User's Manual

Embedded Target for R-Car Virtual Platform
V1.00.00

R-Car Model-Based Development Tool

User's Manual

Target Device

R-Car Family

Renesas Electronics
http://www.renesas.com

Rev.1.00 January 2024

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

How to Use This Manual

Readers	This manual is intended for users who wish to understand the functions of the MATLAB/Simulink and use them to develop application systems.	
Purpose	This manual is intended to educate users about the functions of the model-based development tools, Embedded Target for R-Car Virtual Platform, to be used as reference for developing software systems.	
Composition	The contents of this manual are as follows: Chapter 1 Overview Chapter 2 Installation Chapter 3 Functional Operation Procedure Chapter 4 Points for Caution Chapter 5 Error Messages	
How to Read This Manual	Readers of this manual are assumed to have general knowledge of electricity, logic circuits, and microcontrollers.	
Conventions	Note:	Footnote for items marked with Note in the text.
	Caution:	Information requiring attention.
	Remark:	Supplementary information
	Numeric Representation:	Decimal ... XXXX Hexadecimal ... XXXXH or 0xXXXX
Related Documents		

Licensing

This product uses the FMI based on the following license.

Source code or other data, such as C-header and XML-schema files, that accompany the specification documents are released under the BSD 2-clause license:

Table of Contents

1 Overview

This section provides an overview of functions of Embedded Target for R-Car Virtual Platform (hereafter referred to as ET-VPF).

At the stage of examining algorithms in Model in the Loop Simulation (MILS) used in model-based development, there is no environment to measure the performance of the target device including its peripheral functions.

To estimate the processing time of the peripheral functions in an actual device, it is necessary to combine the code generated from the model with the code for controlling the peripheral functions (hereafter referrers to as peripherals' source code). This is a lot of work for users who do not know the details of the peripheral functions.

ET-VPF generates code for the user algorithm part and generates peripheral code to be implemented on the target from the Simulink model, then runs target code on a partner's R-Car virtual platform (hereafter referred to as VPF) and performs a linked simulation with a Simulink plant model. Renesas call this virtual Hardware in the Loop Simulation (vHILS).

This enables comparative verification of MILS and vHILS (back-to-back test) and performance verification in the early stages of development.

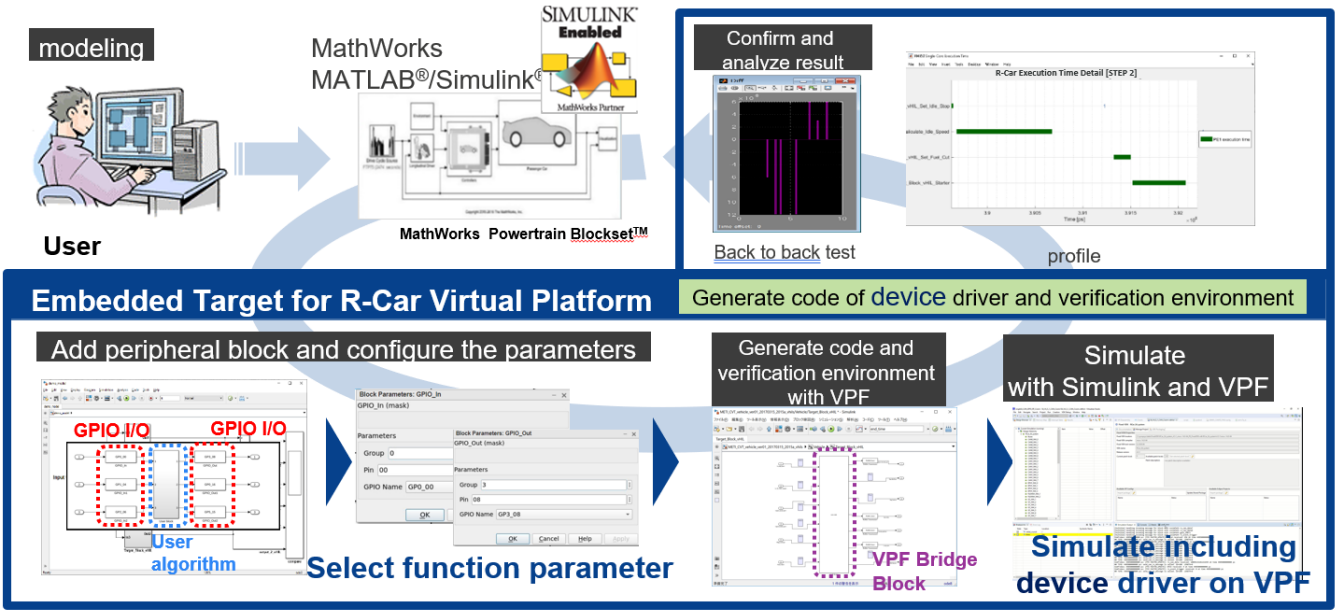


Figure 1-1 Overview of Embedded Target for R-Car S4, R-Car V4H Virtual Platform

1.1 Features

Show below are the features of ET-VPF:

- Automatic construction of vHILSimulation verification environment
  - In vHILSimulation in which MATLAB/Simulink and VDK (R-Car S4, R-Car V4H) are interlinked, a load module generated from Simulink models can be executed on a target device.
  - The following devices are supported.

Table 1-1 Supported devices

Series	Devices
R-Car S4	R-Car S4-8, R-Car S4N-8, R-Car S4-4, R-Car S4N-4
R-Car V4H	R-Car V4H

- Graphical display of the execution state in Subsystem blocks of Simulink models
  - Displaying the execution time for each subsystem during simulation
  - The processing margin of the control cycle for the worst -case execution time during simulation time can be checked.

1.2 Operating Environment

The descriptions below are the system requirements for ET-VPF.

- Hardware environment
  - Operating system: Microsoft Windows® 10 (64-bit)
    * This version only tested on Windows 64-bit
  - Processor: 1 GHz or higher (supporting hyper-threading or Multi-Core CPU)
  - Main memory: 4 GB or more is recommended.

- Software environment
  - MATLAB and Simulink products (from The MathWorks, Inc.)

    Windows 10

MATLAB	9.8 (R2020a)
Simulink	10.1 (R2020a)
Stateflow	10.2 (R2020a)
MATLAB Coder	5.0 (R2020a)
Simulink Coder	9.3 (R2020a)
Embedded Coder	7.4 (R2020a)
Vehicle Network Toolbox	4.4 (R2020a)

  - MEX-file compiler

    Microsoft Visual C++ 2015 compiler (from Microsoft Corporation)
    Reference: System Requirements & Platform Availability
    https://www.mathworks.com/support/sysreq/previous_releases.html

  - GHS compiler

    ccrh850.exe comp_201815 (for supported R-Car S4 device series)
  - ARM compiler (from Advanced RISC Machines Ltd)

    ARM 6.16.1 (for supported R-Car S4, R-Car V4H device series)

  - Functional Mock-up Interface (from Modelica Association Project)

    FMI 2.0 (for supported R-Car S4, R-Car V4H device series)

- Virtualizer Development Kit (from Synopsys)
  VDK T-2022.06-3 (for R-Car S4 device series)
  Q-2020.06 (for R-Car V4H device series)

- Patch RCarV4H EtherAVB Descriptor No32Align RxTx tmp
  Patch 2022/12/22

- MinGW 7.3.0

- MCAL package
  MCAL package Ver19.1.0 (for supported R-Car S4 device series)
  MCAL package Ver19.0.20 (for supported R-Car V4H device series)

- GNU Make 3.81 (for supported R-Car S4, R-Car V4H device series)

Remarks 1. For the MATLAB and Simulink products, an environment is constructed by using option products corresponding to the versions of MATLAB and Simulink being used.
2. When installing MATLAB, it is recommended that the installation folder is changed to something other than the folder for UAC (user account control). Depending on the version of MATLAB in use, if the installation folder is the folder for UAC such as “<system drive>:\Program Files” or “<system drive>:\Program Files (x86)”, a problem such that MEX cannot be built, or the MATLAB path cannot be saved may occur.
3. Can use the upper software versions if the features used for ET-VPF be not changed but Renesas Electronics does not guarantee it.
4. The installation path of ET-VPF package, VDK, MCAL Package, GHS, ARM (support for S4, V4H) cannot contain some special characters (refer to **Table 3-8 The supported special characters for R-Car S4**, **Table 3-13 The supported special characters for R-Car V4H** for more detail).

1.3 License Types and Functions

This section explains the license types.

ET-VPF offers various features to verify algorithms of embedded models. Some features require a specific license, which was registered with Renesas Electronics. This section describes use cases of these features.

Below table shows available operation when you own each license type.

Table 1-2 License definition of ET-VPF

License name		Virtual HILS for R-Car S4x	Virtual HILS for R-Car V4x
Supported device	R-Car S4-4	√	-
	R-Car S4N-4	√	-
	R-Car S4-8	√	-
	R-Car S4N-8	√	-
	R-Car V4H	-	√
Time measurement feature		√	√
Supported build tool	GHS Compiler	√	-
	ARM Compiler	√	√

√ : Can execute
-: Cannot execute.

2 Installation

Start the installer, specify the ET-VPF installation folder, and then execute the installation. For the ET-VPF installation folder, a folder that is targeted for User Account Control (UAC), such as “<system drive>:\Program Files” or “<system drive>:\Program Files (x86)”, cannot be specified.

Remark If installed in a folder targeted for UAC, the ET-VPF cannot be used because, for example, MATLAB path settings cannot be saved.

By extracting the zipped package, the programs, samples, and libraries required for communication between MATLAB and VLAB (hereafter called “MATLAB communication library”) are stored in the following structures.

The following describes the structure of ET-VPF package after installed successfully.

(1) Folder structure under the ET-VPF package folder for R-Car S4 device series

<ET-VPF installation folder>\<version information>\S4x\ ETVPF_package\	ETVPF_include\ BuildTool\	…	Contains the source code and Makefile which is used for ARM compiler and GHS compiler. **Note:** User can modify boot code files in “startup” folder, but Renesas Electronics does not take responsibility for quality.
	ETVPF_include\ETVPF_S_function_block\	…	Contains the source code which used to execute the program of peripherals.
	ETVPF_include\VDK_input\	…	Contains configuration input files for each core (G4MH, CR52).
	ETVPF_include\XML_input\	…	Contains the ARXML input files which use for each module (GPIO, PORT, RS-CANFD, ETHERNET).
	Source code files	…	Contains the list of source code files (include: *.p, *.tlc, make files) which used to execute the program of ET-VPF.

(2) Folder structure under the ET-VPF package folder for R-Car V4H device series

<ET-VPF installation folder>\<version information>\V4x\ ETVPF_package\	ETVPF_include\ETVPF_S_function_block\	…	Contains the source code which is used to execute the program of peripherals.
	ETVPF_include\ARM\	…	Contains the source code and Makefile which used for ARM compiler. Note: User can modify boot code files in “startup” folder, but Renesas Electronics does not take responsibility for quality.
	ETVPF_include\XML_input\	…	Contains the ARXML input files which use for each module (GPIO, PORT, RS-CANFD, ETHERNET).
	Source code files	…	Contains the list of source code files (include: *.p, *.tlc) which used to execute the program of ET-VPF.

Remarks

1. The current <version information> is V1.00.00.
2. For <ET-VPF installation folder>, the user can change them when installing (the user needs to make sure that the installation locations are not limited permission). And the <ET-VPF installation folder> must be in the same <system drive> with the user's workspace to avoid the error occurring when loading the loaded module to R-Car virtual platform.
3. Currently, ET-VPF V1.00.00 of R-Car: support to install ET-VPF package of R-Car device series (R-Car S4x, R-Car V4x).
4. To avoid the warning when downloading the setup file from the website, a zip file for ET-VPF installer ETVPF_<device series>_ <version information>_Setup.7z will be prepared.
5. The <ET-VPF installation folder> must not contain the special characters (refer to **Table 3-8 The supported special characters for R-Car S4**, **Table 3-13 The supported special characters for R-Car V4H**). An error will occur when executing ET-VPF if <ET-VPF installation folder> has the special characters.
76. <device series>: S4x or V4x

2.1 Installation

The following describes the procedure for installing ET-VPF.

(1) Install the ET-VPF package by the below steps:

- **Step 1:** Extract ETVPF_<device series>_ <version information>_Setup.7z file.
- **Step 2:** Double click to ETVPF_<device series>_<version information>_Setup.exe to start installing.
- **Step 3:** Execute step by step to install ET-VPF package as the following figure.

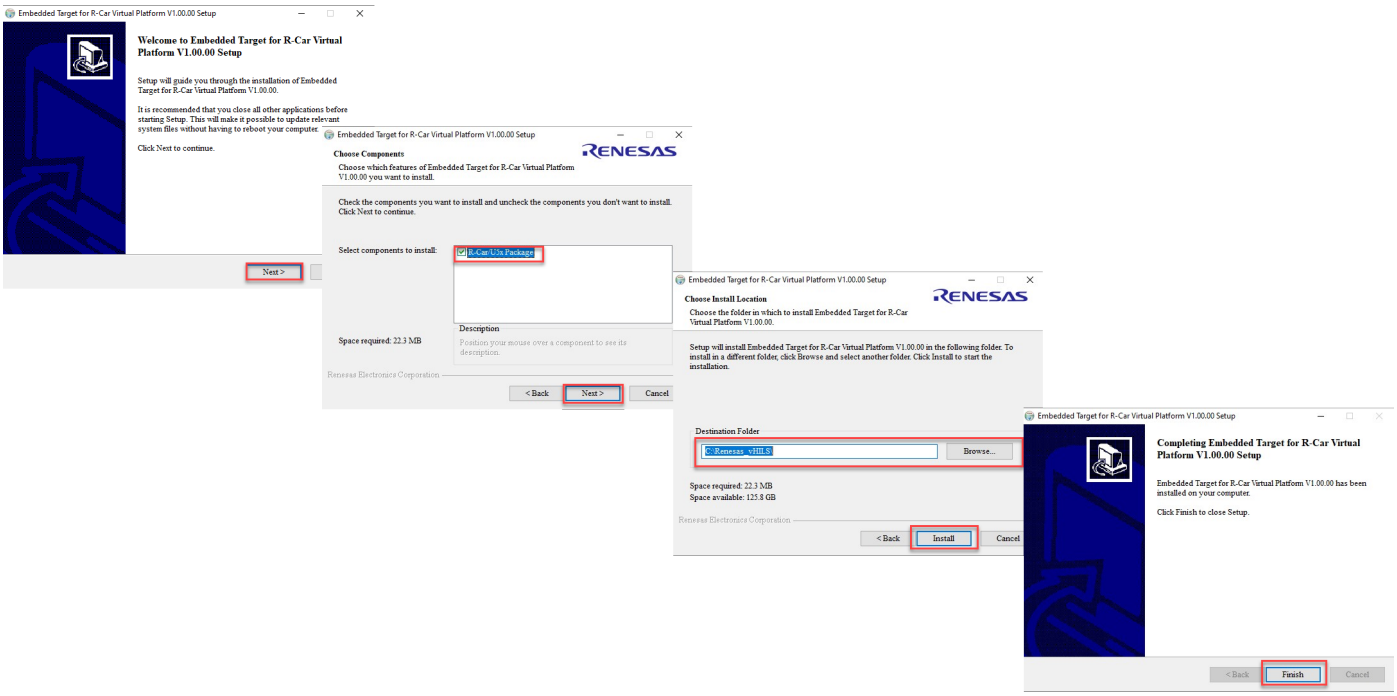


Figure 2-1 Installation Procedure

(2) After installing the ET-VPF package, the user must set a path for ET-VPF package. Start MATLAB, and then add the folder of ET-VPF on the [Set Path] dialog box.

There are two necessary paths which need to set to MATLAB:

- **ET-VPF include:** …\ETVPF_package\ETVPF_include
- **ET-VPF S-Function:** …\ETVPF_package\ETVPF_include\ETVPF_S_function_block

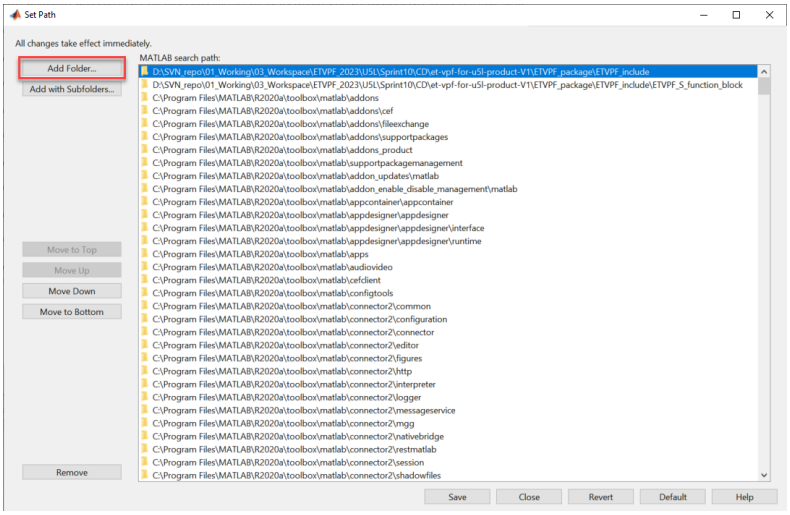


Figure 2-2 Adding the ET-VPF folder

(3) Register MATLAB Automation Server.
Execute the following command from the MATLAB command window to specify the current in-use MATLAB version as the Automation Server.
Here ">>" denotes the command prompt and "[Enter]" denotes entry of the Enter key.

```
>> regmatlabserver [Enter]
```

Remarks 1. Open MATLAB under administration privilege when executing this command.
2. If you change the in-use MATLAB version, execute this command again.

(4) Register the license.
Add the ET-VPF license by license manager (refer License Manager V2.05.00 User's manual).

(5) Setup the MATLAB Simulink Library Browser
Execute the following command from the MATLAB command window to open ETVPF package on the MATLAB Simulink Library Browser to choose the expected S-function blocks of peripherals.
Here ">>" denotes the command prompt and "[Enter]" denotes entry of the Enter key.

```
>> setup_etvpf_lib [Enter]
```

Use GNU Make to compile the make script in the MCAL source. Please download and install the following tools from the website.
GNU Make (For supported R-Car S4 and R-Car V4H device series)
Make for Windows (sourceforge.net)
Remark We need to set a window environment variable with the name "GNUMAKE", and the value is path of GNU Make that contains "make.exe".

2.2 Uninstallation

The following describes the procedure for uninstalling ET-VPF.

(1) Start MATLAB, and then remove the folder of ET-VPF (include both two paths described in section **Installation**) on the [Set Path] dialog box.

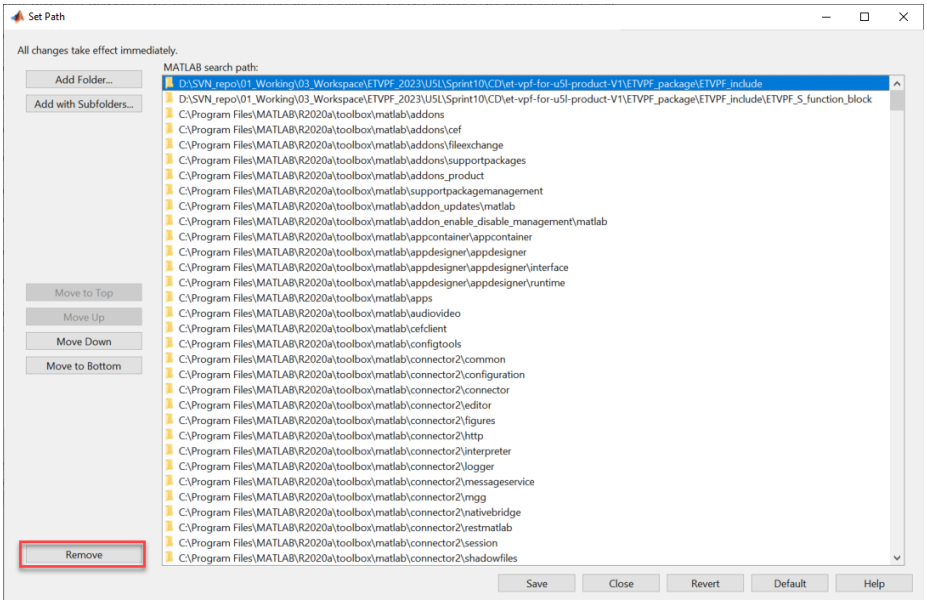


Figure 2-3 Removing the ET-VPF folder

(2) Execute Uninst_ETVPF_<device series>_<version information>.exe in the ET-VPF installation folder as below steps:

- **Step 1:** Double click to Uninst_ETVPF_<device series>_<version information>.exe to start uninstalling ET-VPF package.
- **Step 2:** Execute step by step to uninstall ET-VPF package as the following figures.

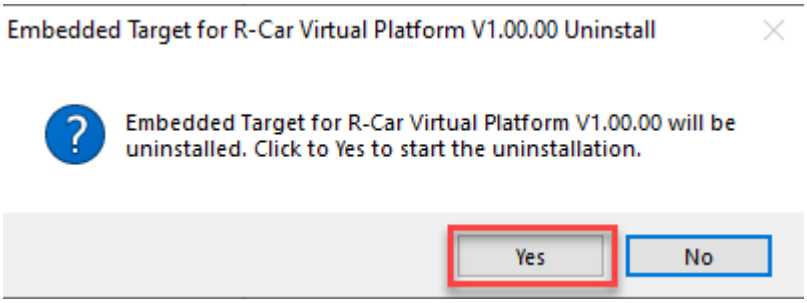


Figure 2-4 Uninstallation Confirmation Message

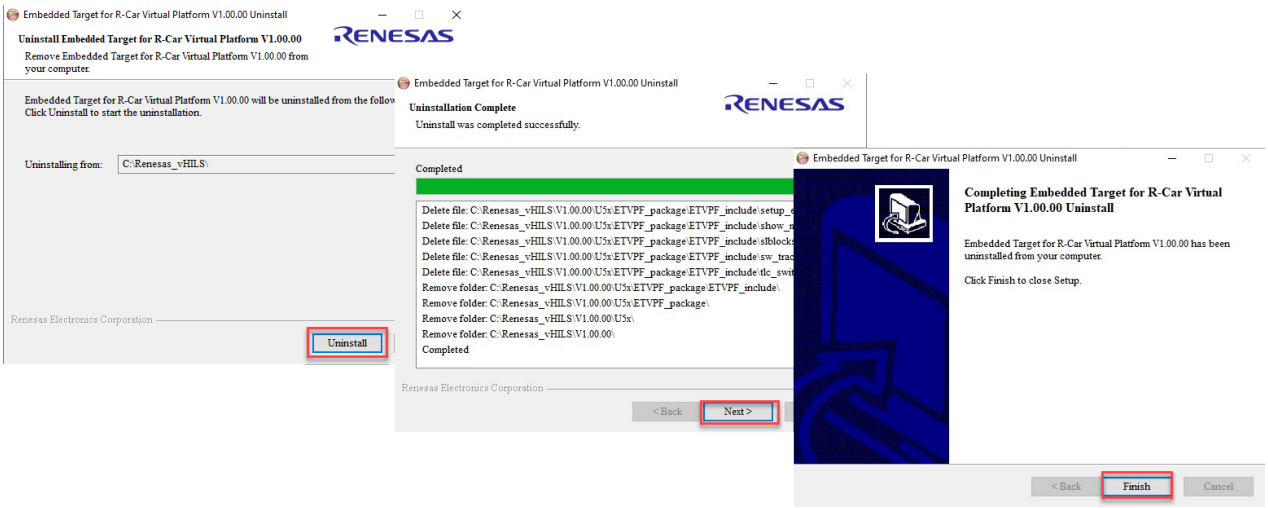


Figure 2-5 Uninstallation Procedure

Remark If (2) is performed prior to (1) above, a warning will be displayed the next time MATLAB is started.

(3) Uninstall the ET-VPF category in Simulink Library Browser window as below step:
Execute the following command from the MATLAB command window to remove ETVPF category in the MATLAB Simulink Library Browser window.
Here ">>" denotes the command prompt and "[Enter]" denotes entry of the Enter key.

```
>> refresh (LibraryBrowser.LibraryBrowser2) [Enter]
```

3 Functional Operation Procedure

This chapter describes the functions provided by ET-VPF.

3.1 Overview

ET-VPF provides functions to generate a vHILS environment and to verify algorithms. ET-VPF generates a vHILS environment in cooperation with Embedded Coder.

The following describes the main procedure of ET-VPF:

- Set configuration parameters.
- Clean existing files, objects.
- Generate source code files.
  - Generate source files of target device.
  - Generate C source files.
  - Generate make file and config file.
  - Generate application make file.
  - Generate VDK load module file.
  - Generate python files.
  - Generate OSTM_define.h
- Start-up VDK then runs the target script.
- Wait until the co-sim finishes and verify the result.

About S-Function block of peripherals (hereafter referred to as peripheral block), there are blocks included in “etvpf_lib.slx” file:

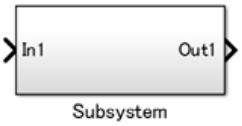
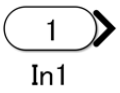
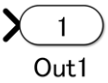
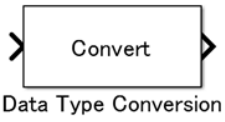
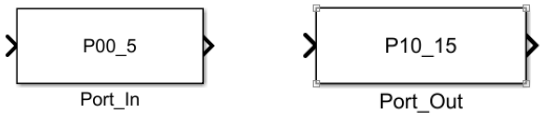
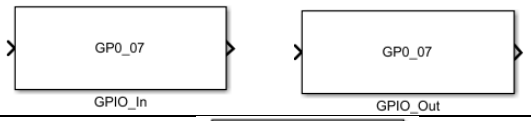
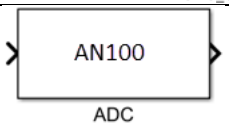
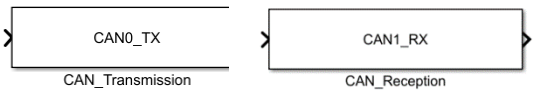
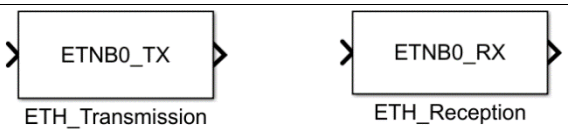
- R-Car S4, R-Car V4H: GPIO, RS-CANFD, ETHERNET

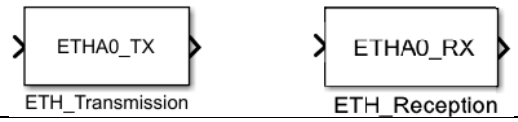
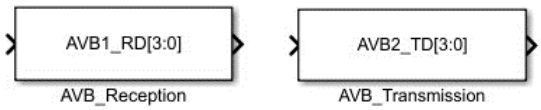
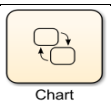
ET-VPF will support generating a vHILS environment, verify algorithms, configure settings, generate source code for peripherals automatically.

- R-Car S4, R-Car V4H: GPIO, RS-CANFD, ETHERNET

The following table shows the list of the blocks that can be placed in the same layer as the measurement target block (hereafter referrerd to as usable blocks).

Table 3-1 List of usable blocks

No	Usable blocks	Configurable blocks
1	Subsystem	
2	Inport (*1)	
3	Outport	
4	Mux	
5	Demux	
6	Data Type Conversion	
7	PORT	
8	GPIO	
9	ADC	
10	RS-CANFD	
11	RS-CANFD (*2)	
12	ETHERNET AVB-IF (for R-Car S4)	

	ETHERNET RSW2 (for R-Car S4)	
	ETHERNET (for R-Car V4H)	
13	Chart	

(*1) This block can be created by right clicking and then dragging the original Inport block and then selecting “Duplicate Inport” as the Inport shadow. This causes confusion and Renesas Electronics does not recommend users to use the Inport shadow block.

(*2) This block is only used for R-Car V4H, because in HW Pin Name is described differently from other devices.

3.2 S-Function block of peripherals

The following describes the S-Function blocks of peripherals.

- For R-Car S4, R-Car V4H: GPIO, RS-CANFD, ETHERNET

how to create, set and generate C code for these S-Function blocks.
In addition, the MATLAB Simulink Library Browser supports ETVPF package, which contained the S-function blocks of peripherals to add into the model.

Remarks 1. The S-Function blocks of peripherals are different with each device series. Therefore, the user must select the correct peripherals’ settings with the current device series. The sample S-Function blocks of each device series are also included inside the “etvpf_lib.slx” file.

2. About the peripherals’ settings used for R-Car, only use the following settings.

a) For R-Car V4H: Only can use GPIO (GPIO), RS-CANFD (support use CANFD unit 0 with 8 channels, Sending/receiving Data frame, Standard ID, the remote frame, extended ID, receiving filter, Support CAN Port I/O and Vehicle Network Toolbox), ETHERNET (support use AVB unit 0, 1, 2).

b) For R-Car S4 Only can use GPIO (GPIO), RS-CANFD (support use CANFD unit 0, 1 with 16 channels), ETHERNET (include two functions, ETHERNET AB-IF with 1 unit for G4MH core and ETHERNET RSW2 with 3 unit for CR52 core)

3. The S-Function blocks of peripherals are prepared for the largest device. If using the smaller device, please select the function, port name which is implemented for this device.

3.2.1 PORT peripheral

The following describes about the main features of Port peripheral.

- **For Port_In block:**
- This block receives the data (boolean) as input from MATLAB, then it transfers data to User Algorithm (boolean value).
- **For Port_Out block:**
- This block receives the data (boolean) as input from User Algorithm, then it transfers data to MATLAB (boolean value).

Figure 3-1 S-Function block of Port

The following describes the User Interface of Port S-Function block that supports user select and change ports easier during model setting. The User Interface also has two types corresponding to two types of Port S-Function block (Port_In and Port_Out).

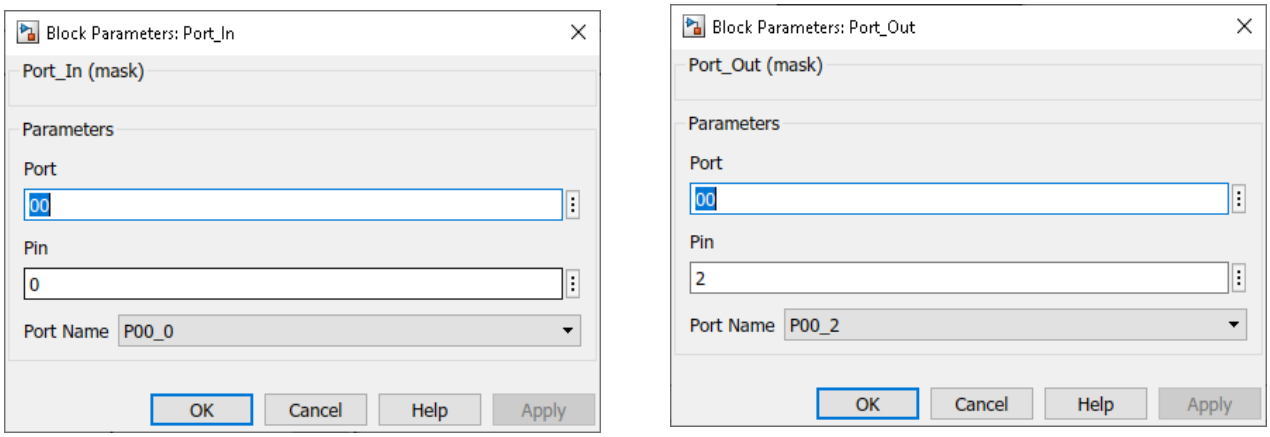


Figure 3-2 User Interface of Port S-Function block

The User Interface of Port S-Function block includes “Port Name”, “Port” and “Pin” parameters.

The purpose of these parameters is to specify the target port name of Port S-Function block. When building the model, these parameters will generate at the same time as the input data.

After changing the port name, the parameters (Port, Pin) and displaying name of S-Function block will be changed automatically based on the current port name.

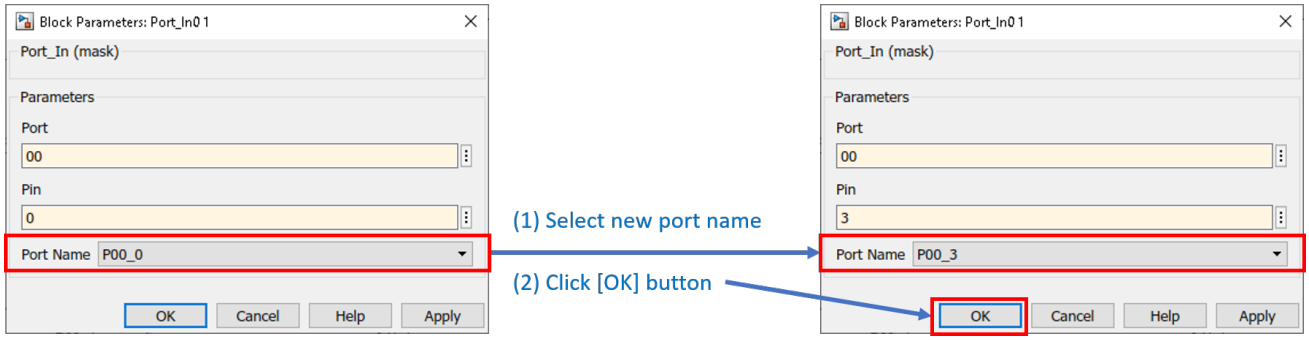


Figure 3-3 Changing port name of Port S-Function block

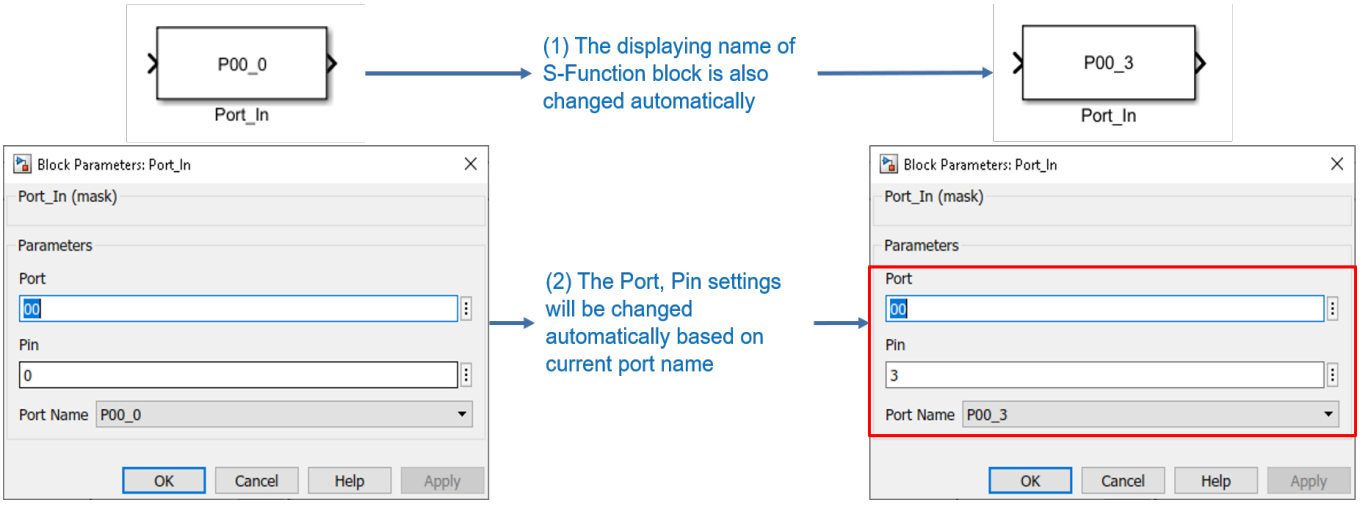


Figure 3-4 The changing result of Port S-Function block

3.2.2 GPIO peripheral

The following describes the main features of Port peripheral.

- **For GPIO_In block:**
  - Get data from MATLAB and send it to GPIO in VDK.
  - Send the received value to the user algorithms.
- **For GPIO_Out block:**
  - Set the value from user algorithms to the target GPIO name.
  - Send the value of GPIO to the MATLAB (output of the MATLAB Co-simulation).

Figure 3-5 S-Function block of GPIO

The following describes the User Interface of GPIO S-Function block that supports user selection and changes GPIOs easier during model setting. The User Interface also has two types, which correspond to two types of GPIO S-Function blocks (GPIO In and GPIO Out).

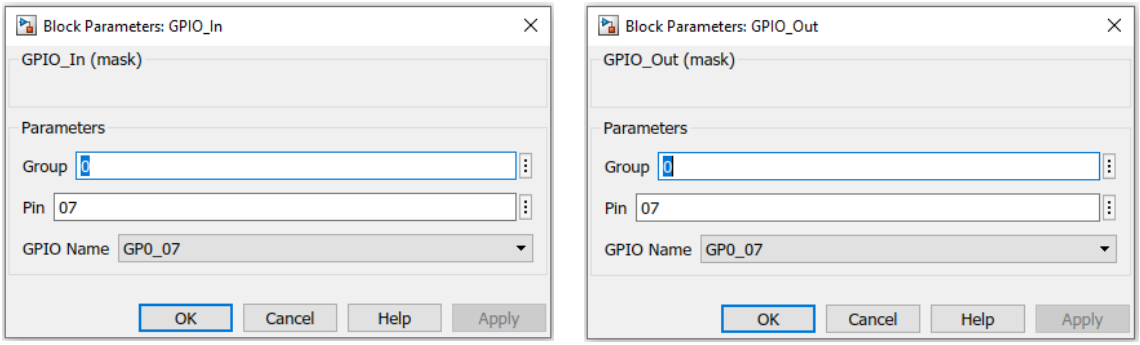


Figure 3-6 User Interface of GPIO S-Function block

The User Interface of GPIO S-Function block includes “GPIO Name”, “Group” and “Pin” parameters.

The purpose of these parameters is to specify the target port name of GPIO S-Function block. When building the model, these parameters will generate at the same time as the input data.

After changing the port name, the parameters (Group, Pin) and displaying name of S-Function block will be changed automatically based on the current GPIO name.

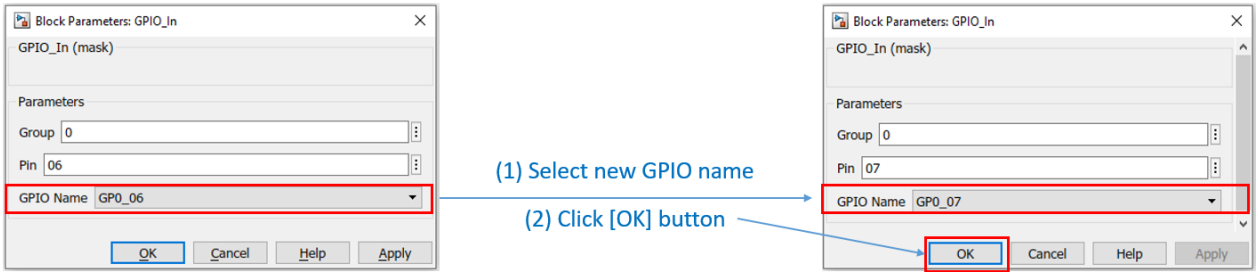


Figure 3-7 Changing GPIO name of GPIO S-Function block

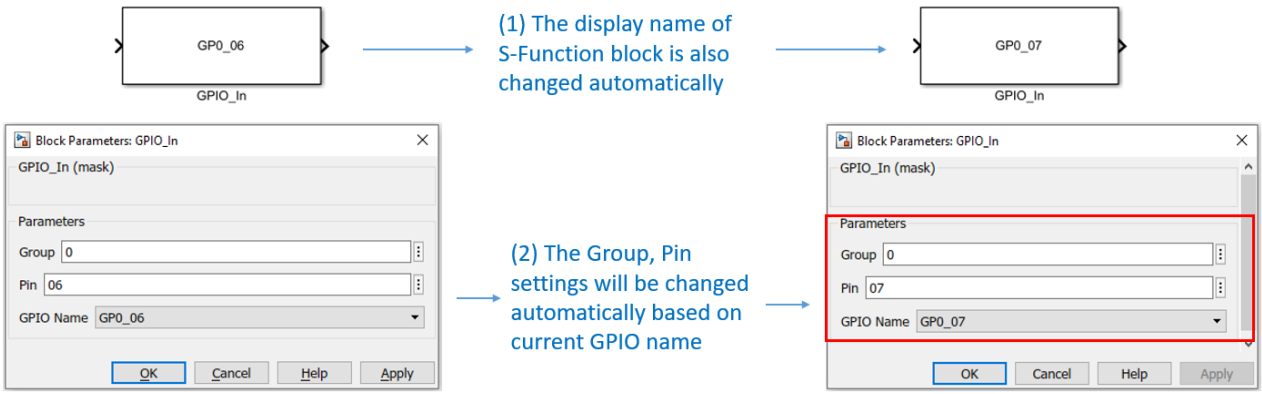


Figure 3-8 The changing result of GPIO S-Function block

3.2.3 ADC peripheral

The following describes the main features of ADC peripheral.

- This block receives the data (double) as input from MATLAB, then it transfers data to User Algorithm (double value).

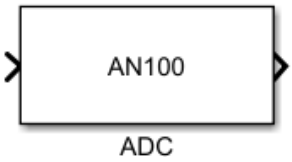


Figure 3-9 S-Function block of ADC

The following describes the User Interface of ADC S-Function block that supports users selecting and changing ADC ports easier during model settings.

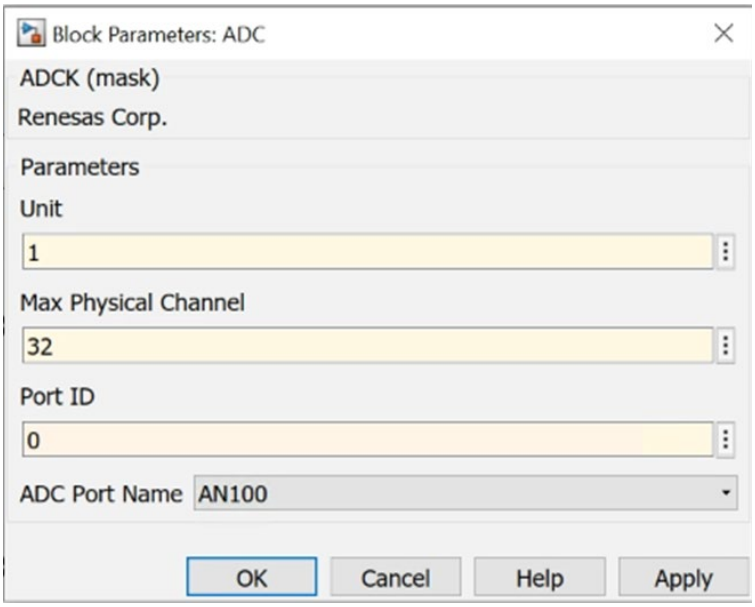


Figure 3-10 User Interface of ADC S-Function block

The User Interface of ADC S-Function block includes “ADC Port Name”, “Unit”, “Max Physical Channel” and “Port ID” parameters.

The purpose of these parameters is to specify the target port name of ADC S-Function block. When building the model, these parameters will generate at the same time as the input data.

After changing the ADC port name, the parameters (Unit, Max Physical Channel, Port ID) and the name of the S-Function block will be changed automatically based on the current ADC port name.

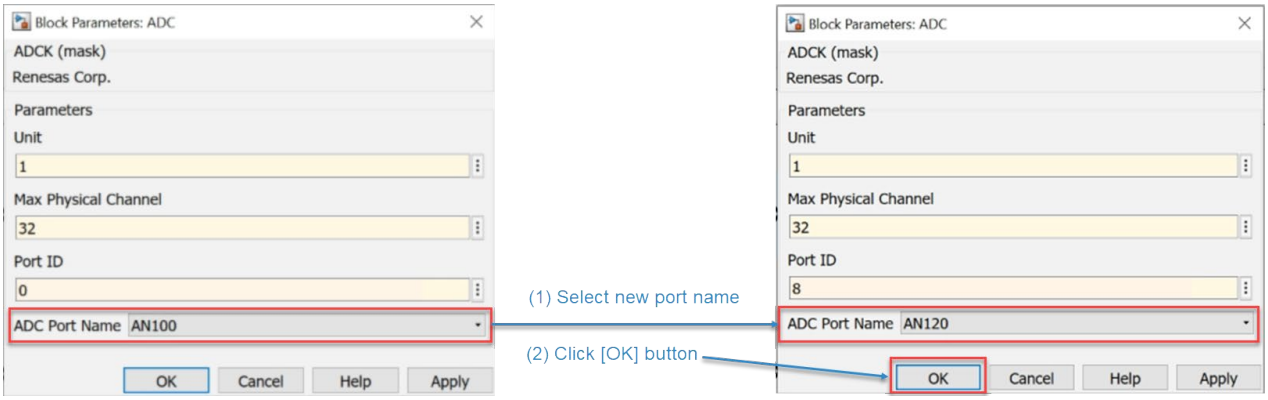


Figure 3-11 Changing port name of ADC S-Function block

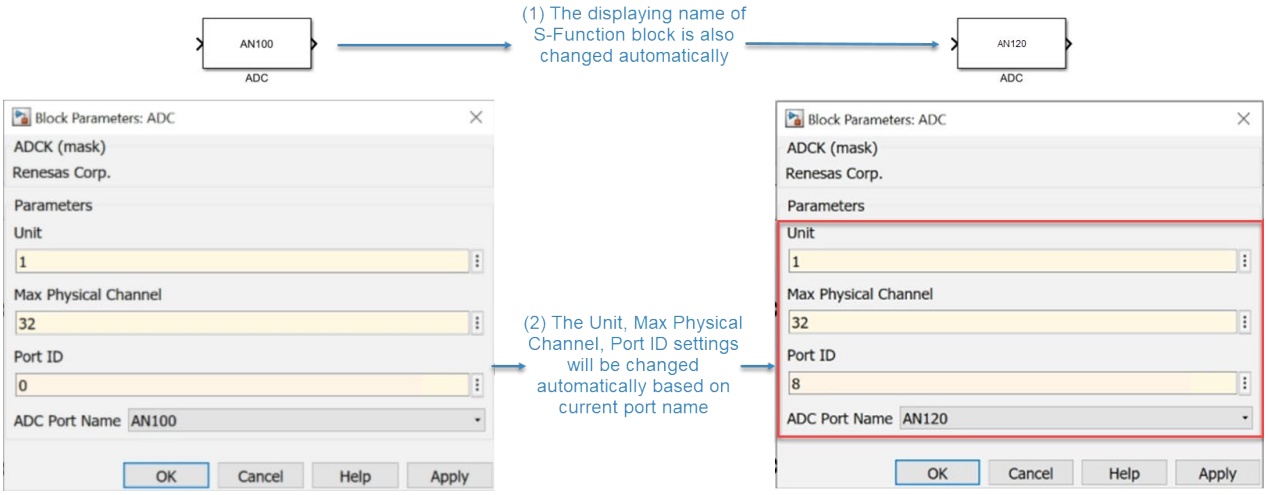


Figure 3-12 The changing result of ADC S-Function block

3.2.4 RS-CANFD peripheral

The following describes the main features of RS-CANFD peripheral.

- **For CAN_Transmission block:**
  - Get array bytes (dynamic length) from User algorithm, transmit message through RS-CANFD port, then output to MATLAB (bytes array data).
- **For CAN_Reception block:**
  - Get array bytes (dynamic length) from MATLAB, transmit through RS-CANFD port, output to user algorithm (bytes array data).

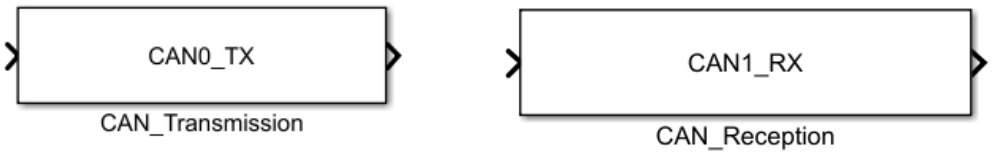


Figure 3-13 S-Function block of RS-CANFD for R-Car S4

Figure 3-14 S-Function block of RS-CANFD for R-Car V4H

The following describes the User Interface of RS-CANFD S-Function block that support user select and change RS-CANFD ports easier during model setting. The User Interface also have two types be corresponding to two RS-CANFD S-Function blocks (CAN_Transmission and CAN_Reception).

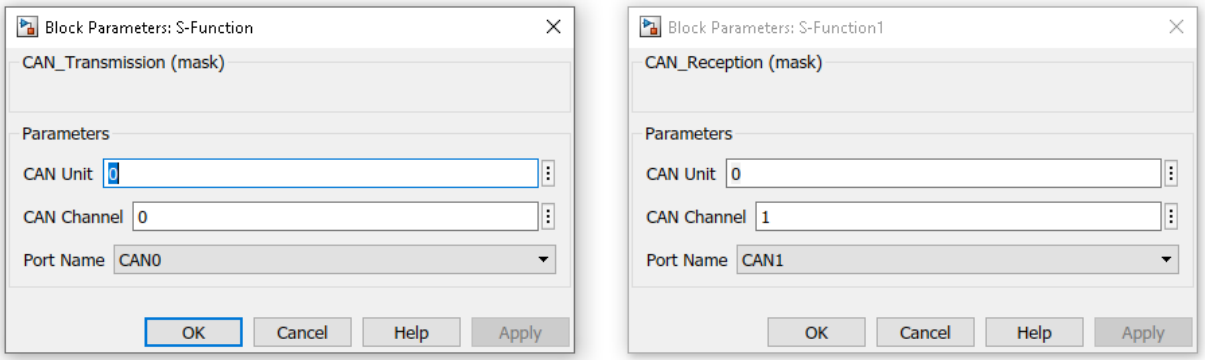


Figure 3-15 User Interface of RS-CANFD S-Function block for R-Car S4

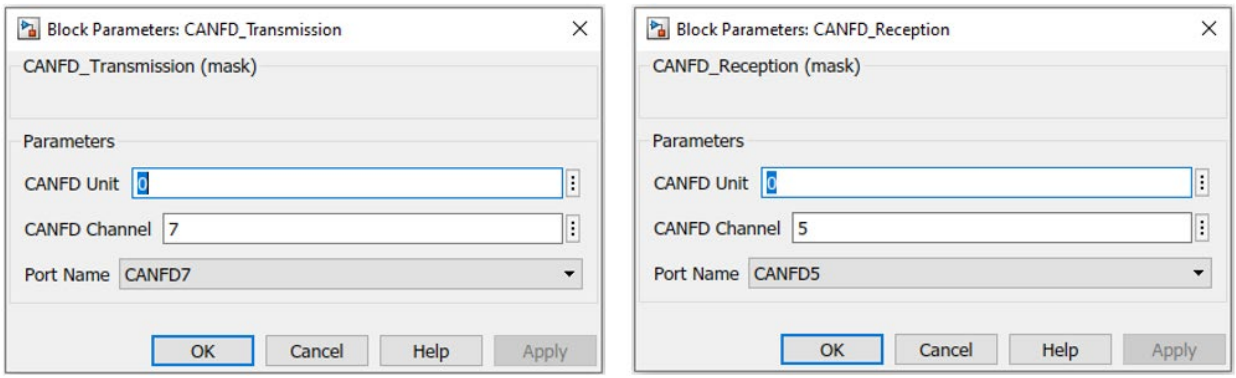


Figure 3-16 User Interface of RS-CANFD S-Function block for R-Car V4H

For R-Car V4H:

- The User Interface of RS-CANFD S-Function block includes “CANFD Unit”, “CANFD Channel”, “Port Name” parameters. The “CANFD Unit” and “CANFD Channel” would be generated automatically after changing the “Port Name”.

The purpose of these parameters is to specify the target port name of RS-CANFD S-Function block. When building the model, these parameters will generate at the same time as the input data.

For the message ID: ET-VPF prepared the CAN ID array (with the default ID message value as CAN unit number) for each CAN unit. Users can change it as they demand. ET-VPF support gets the ID and extension ID from the CAN Pack and CAN Unpack block for each unit and set to the CAN ID array when generating.

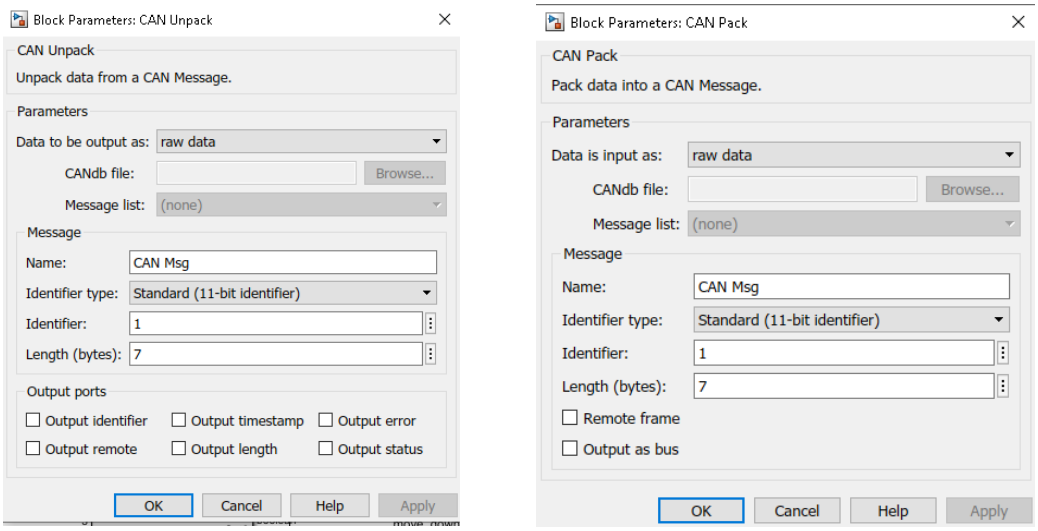


Figure 3-17 Information from CAN Pack and CAN Unpack

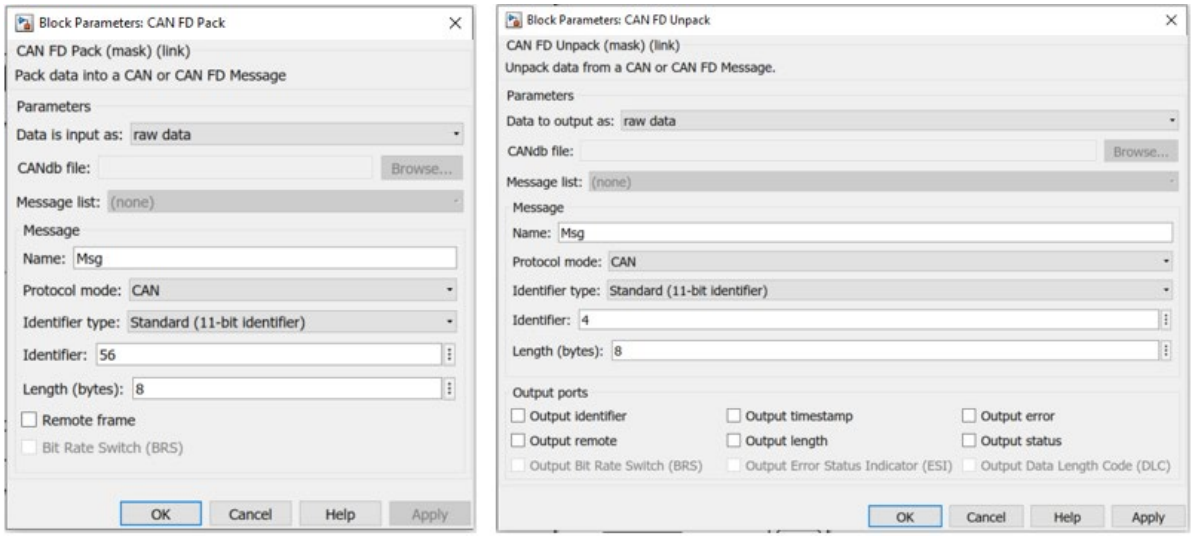


Figure 3-18 Information from CANFD Pack and CANFD Unpack

- Model structure between MATLAB and VDK:

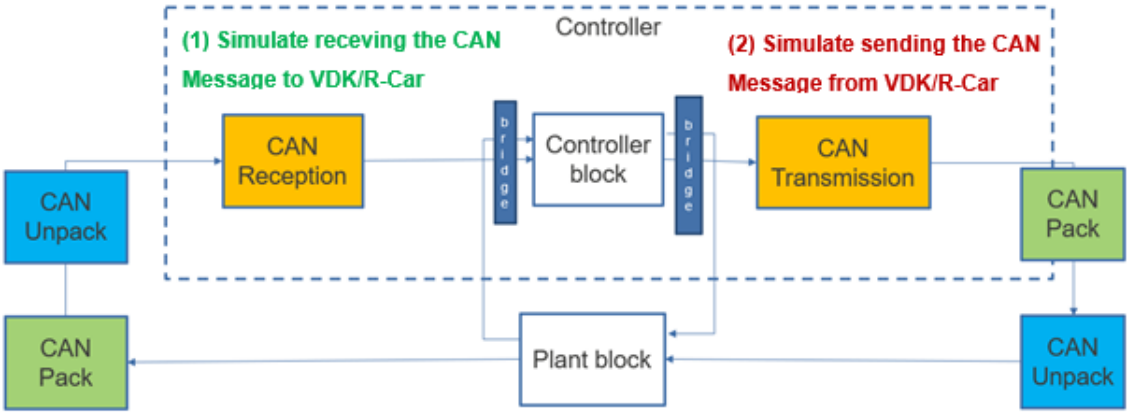


Figure 3-19 Model structure for using RS-CANFD for R-Car S4

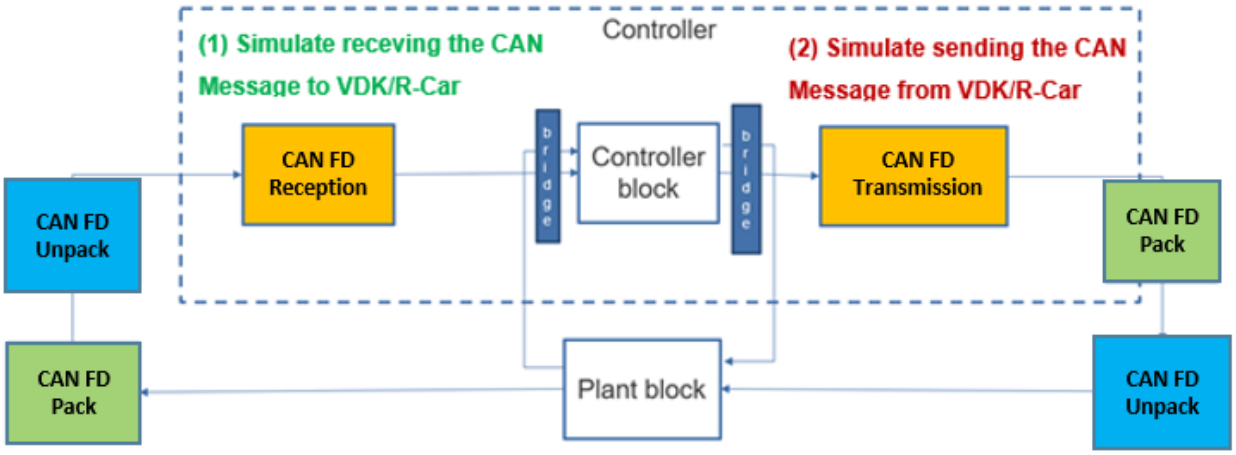


Figure 3-20 Model structure for using RS-CANFD For R-Car V4H

User must connect CAN Reception block and CAN transmission block and CAN Pack/Unpack same as above structure for conduct ETVPF with RS-CANFD.

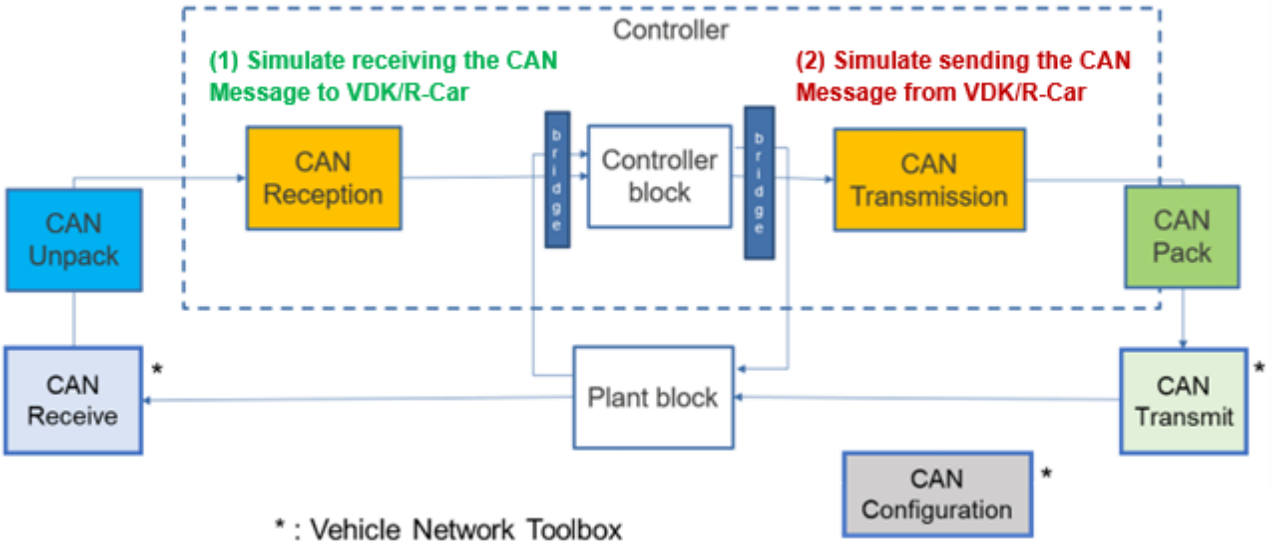


Figure 3-21 Model structure for using RS-CANFD for R-Car S4 with Vehicle Network Toolbox

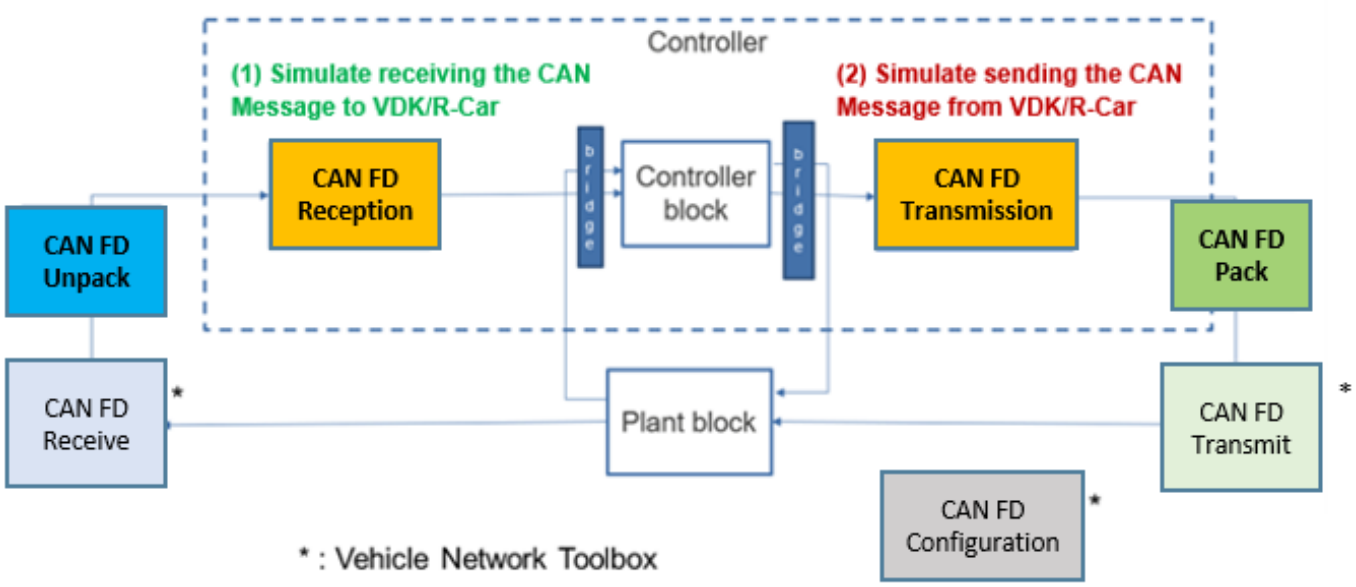


Figure 3-22 Model structure for using RS-CANFD for R-Car V4H with Vehicle Network Toolbox

Remarks

In the specification of CAN Driver AUTOSAR CP R19-11, the Can module does not support CAN remote frames.
Link: https://www.autosar.org/fileadmin/standards/R19-11/CP/AUTOSAR_SWS_CANDriver.pdf (Page 20).

Table 3-2 The Port Name is used for RS-CANFD

Device Series	RS-CANFD	Port Name
R-Car S4	CAN0TX	GP7_00
	CAN0RX	GP7_01
	CAN1TX	GP7_02
	CAN1RX	GP7_03
	CAN2TX	GP7_04
	CAN2RX	GP7_05
	CAN3TX	GP7_06
	CAN3RX	GP7_07
	CAN4TX	GP7_08
	CAN4RX	GP7_09
	CAN5TX	GP7_10
	CAN5RX	GP7_11
	CAN6TX	GP7_12
	CAN6RX	GP7_13
	CAN7TX	GP7_14
	CAN7RX	GP7_15
	CAN8TX	GP7_16
	CAN8RX	GP7_17
	CAN9TX	GP7_18
	CAN9RX	GP7_19
	CAN10TX	GP7_20
	CAN10RX	GP7_21
	CAN11TX	GP7_22
	CAN11RX	GP7_23
	CAN12TX	GP7_24
	CAN12RX	GP7_25
	CAN13TX	GP7_26
	CAN13RX	GP7_27
	CAN14TX	GP7_28
	CAN14RX	GP7_29
	CAN15TX	GP7_30
	CAN15RX	GP7_31
R-Car V4H	CANFD0_TX	GP2_10
	CANFD0_RX	GP2_11
	CANFD1_TX	GP2_00
	CANFD1_RX	GP2_01
	CANFD2_TX	GP2_12
	CANFD2_RX	GP2_13
	CANFD3_TX	GP2_14
	CANFD3_RX	GP2_15
	CANFD4_TX	GP2_16
	CANFD4_RX	GP2_17
	CANFD5_TX	GP2_02
	CANFD5_RX	GP2_03
	CANFD6_TX	GP2_07
	CANFD6_RX	GP2_08
	CANFD7_TX	GP2_18
	CANFD7_RX	GP2_19

3.2.5 ETHERNET peripheral

The following descriptions describe about the main features of ETHERNET peripheral.

For R-Car S4:

- **Transmission block:**
  - Get array bytes (dynamic length) from user algorithm and transmit data through ETHERNET port. Then output to MATLAB (bytes array data).
- **ETHERNETn Reception block:**
  - Get array bytes (dynamic length) from MATLAB, transmit through ETHERNET port, output to user algorithm (bytes array data).

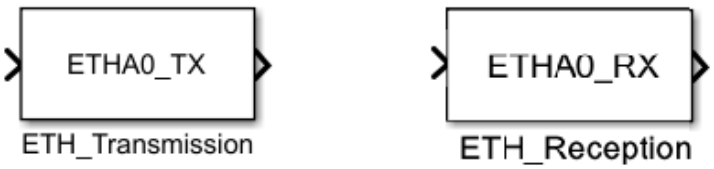


Figure 3-23 S-Function block of ETHERNET

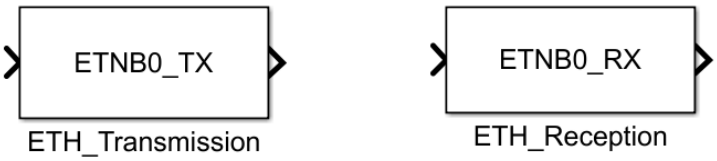


Figure 3-24 S-Function block of ETHERNET

The following descriptions describe how the User Interface of ETHERNET S-function block is used to support user selection and change ETHERNET ports easier during the setting model. The User Interface includes two types, which are corresponding to two ETHERNET S-function blocks (ETH_Transmission and ETH_Reception).

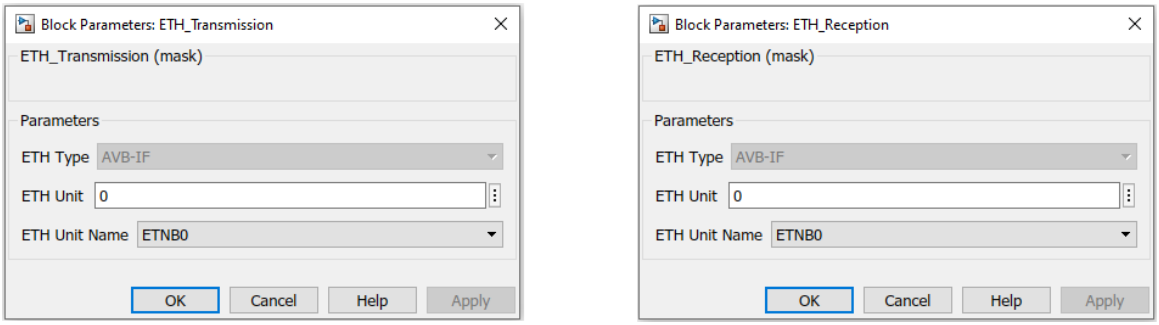


Figure 3-25 User Interface of ETHERNET S-Function block

The User Interface of ETHERNET S-Function block includes “ETH Type”, “ETH Unit” and "ETH Unit Name" parameters.

The purpose of these parameters is to specify the target port name of ETHERNET S-Function block. When building the model, these parameters will be generated at the same time as the input data.

After changing the "ETH Unit Name" of ETHERNET, the displaying name of S-function block will be changed automatically based on the current selecting "ETH Unit Name".

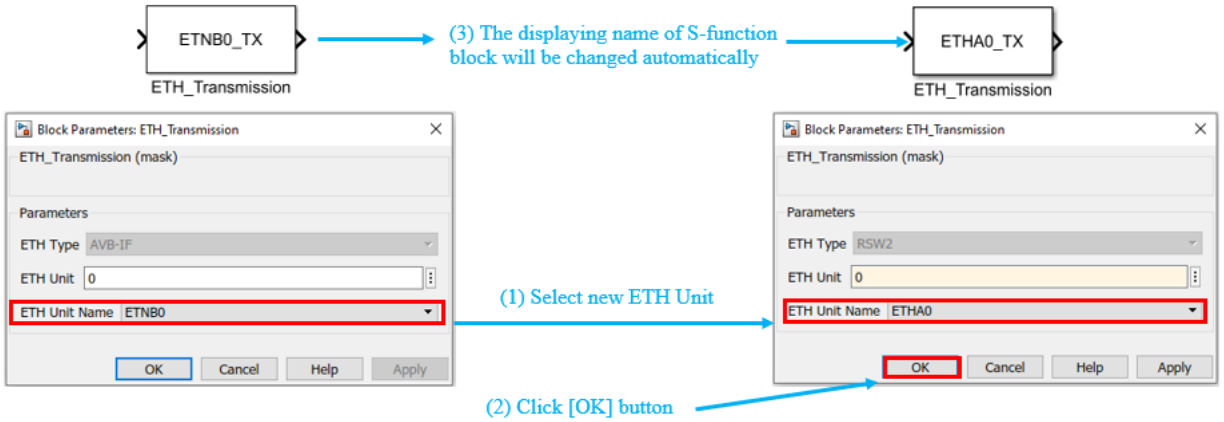


Figure 3-26 The changing result of ETHERNET S-Function block

Table 3-3 The Port Name is used for ETHERNET of device series R-Car S4

Device Series	ETHERNET	ETHERNET	Port Name
R-Car S4	ETH Unit Name	ETNB0_TX (*1) ETNB0_RX	ETNB0TXD0 (GP5_19) ETNB0TXD1 (GP5_16) ETNB0TXCLK (GP5_15) ETNB0TXEN (GP5_18) ETNB0RXD0 (GP5_11) ETNB0RXD1 (GP5_08) ETNB0RXCLK (GP5_12) ETNB0MD (GP5_02) ETNB0WOL (GP5_03) ETNB0LINKSTA (GP5_04) ETNB0MDC (GP5_05)
		ETHA0_TX (*1) ETHA0_RX ETHA1_TX ETHA1_RX ETHA2_TX ETHA2_RX	TSN0_MDIO (GP3_02) TSN0_MDC (GP3_04) TSN0_LINK (GP3_08) TSN0_PHY_INT (GP3_10) TSN0_MAGIC (GP3_12) TSN0_AVTP_PPS GP3_16) TSN0_AVTP_MATCH (GP3_17) TSN0_AVTP_CAPTURE (GP3_18) TSN1_MDIO (GP3_00) TSN1_MDC (GP3_05) TSN1_LINK (GP3_06) TSN1_PHY_INT (GP3_11) TSN1_AVTP_PPS (GP3_13) TSN1_AVTP_MATCH (GP3_14) TSN1_AVTP_CAPTURE (GP3_15) TSN2_MDIO (GP3_01) TSN2_MDC (GP3_03) TSN2_LINK (GP3_07) TSN2_PHY_INT (GP3_09)

Remarks

*1… For R-Car S4 support ETHERNET AVB-IF with 1 unit for G4MH core and support ETHERNET RSW2 with 3 unit for CR52 core.

For R-Car V4H:

- **AVB (*1) Transmission block:**
  - Get array bytes (dynamic length) from user algorithm and transmit data through ETHERNET port. Then output to MATLAB (bytes array data).
- **AVB Reception block:**
  - Get array bytes (dynamic length) from MATLAB, transmit through ETHERNET port, output to user algorithm (bytes array data).

Figure 3-27 S-Function block of ETHERNET

The following descriptions describe how the User Interface of ETHERNET S-function block is used to support user selection and change ETHERNET ports easier during the setting model. The User Interface includes two types, which correspond to two ETHERNET S-function blocks (AVB_Transmission and AVB_Reception).

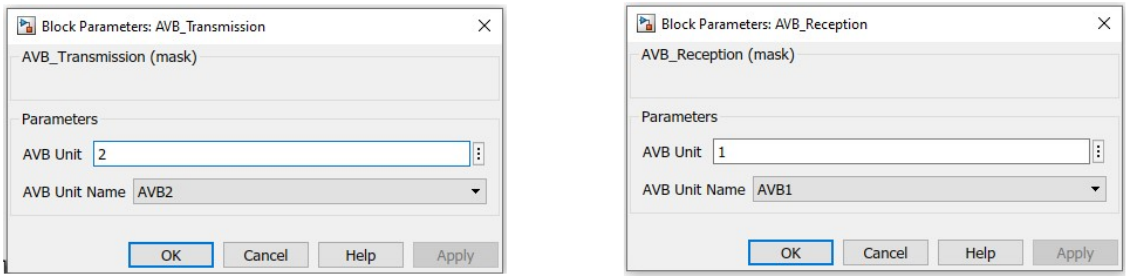


Figure 3-27 User Interface of ETHERNET S-Function block

The User Interface of ETHERNET S-Function block includes “AVB unit” and “AVB Unit name” parameters.

The purpose of these parameters is to specify the target port name of ETHERNET S-Function block. When building the model, these parameters will be generated at the same time as the input data.

After changing the unit of ETHERNET, the displaying name of S-function block will be changed automatically based on the current selecting ETHERNET unit.

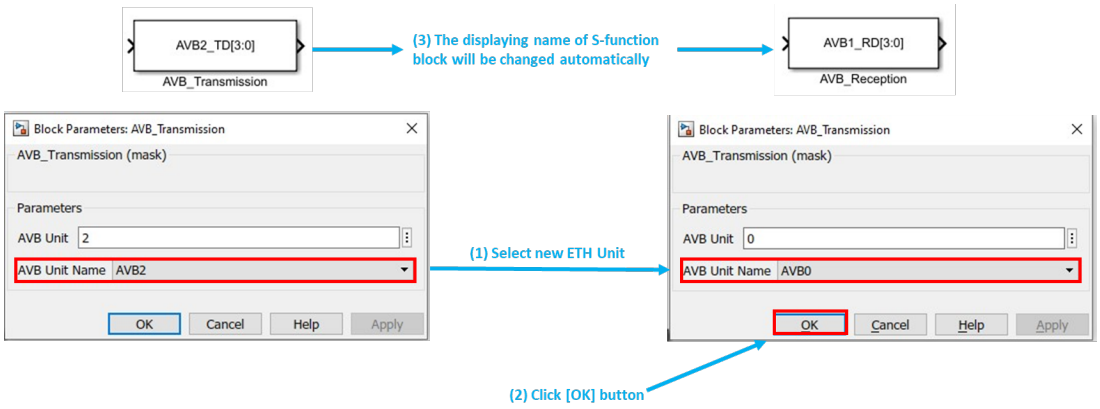


Figure 3-27 The changing result of ETHERNET S-Function block

Remark:

(*1) … AVB (Audio Video Bridging) is an extension of the Ethernet standard. It is a set of standards that provide the ability to transmit real-time audio and video over Ethernet networks. AVB includes key features such as Stream Reservation Protocol (SRP), Timing and Synchronization for Time-Sensitive Applications (gPTP), and Quality of Service (QoS).

Table 3-4 The Port Name is used for ETHERNET

Device Series	ETHERNET	Port Name
R-Car V4H	AVB0_TD[3:0] AVB0_RD[3:0]	AVB0_TD3 (GP7_03) AVB0_LINK (GP7_04) AVB0_PHY_INT (GP7_05) AVB0_TD2 (GP7_06) AVB0_TD1 (GP7_07) AVB0_RD3 (GP7_08) AVB0_TXCREFCLK (GP7_09) AVB0_MAGIC (GP7_10) AVB0_TD0 (GP7_11) AVB0_RD2 (GP7_12) AVB0_MDC (GP7_13) AVB0_MDIO (GP7_14) AVB0_TXC (GP7_15) AVB0_TX_CTL (GP7_16) AVB0_RD1 (GP7_17) AVB0_RD0 (GP7_18) AVB0_RXC (GP7_19) AVB0_RX_CTL (GP7_20)
	AVB1_TD[3:0] AVB1_RD[3:0]	AVB1_MDIO (GP6_00) AVB1_MAGIC (GP6_01) AVB1_MDC (GP6_02) AVB1_PHY_INT (GP6_03) AVB1_LINK (GP6_04) AVB1_TXC (GP6_06) AVB1_TX_CTL (GP6_07) AVB1_RXC (GP6_08) AVB1_RX_CTL (GP6_09) AVB1_TD1 (GP6_12) AVB1_TD0 (GP6_13) AVB1_RD1 (GP6_14) AVB1_RD0 (GP6_15) AVB1_TD2 (GP6_16) AVB1_RD2 (GP6_17) AVB1_TD3 (GP6_18) AVB1_RD3 (GP6_19) AVB1_TXCREFCLK (GP6_20)
	AVB2_TD[3:0] AVB2_RD[3:0]	AVB2_LINK (GP5_03) AVB2_PHY_INT (GP5_04) AVB2_MAGIC (GP5_05) AVB2_MDC (GP5_06) AVB2_TXCREFCLK (GP5_07) AVB2_TD3 (GP5_08) AVB2_RD3 (GP5_09) AVB2_MDIO (GP5_10) AVB2_TD2 (GP5_11) AVB2_TD1 (GP5_12) AVB2_RD2 (GP5_13) AVB2_RD1 (GP5_14) AVB2_TD0 (GP5_15) AVB2_TXC (GP5_16) AVB2_RD0 (GP5_17) AVB2_RXC (GP5_18) AVB2_TX_CTL (GP5_19) AVB2_RX_CTL (GP5_20)

3.3 Executing Virtual Hardware in the Loop Simulation R-Car S4

The following describes how to generate a vHILS environment necessary for Simulator Processor in the Loop Simulation (hereafter referred to as vHILS) by ET-VPF.

3.3.1 Embedded sample model

Firstly, the user needs to open ETVPF package of the MATLAB Simulink Library Browser to choose the expected S-function blocks of peripherals.

- The ETVPF package is in MATLAB Simulink Library Browser, which contains the S-function blocks of peripherals.

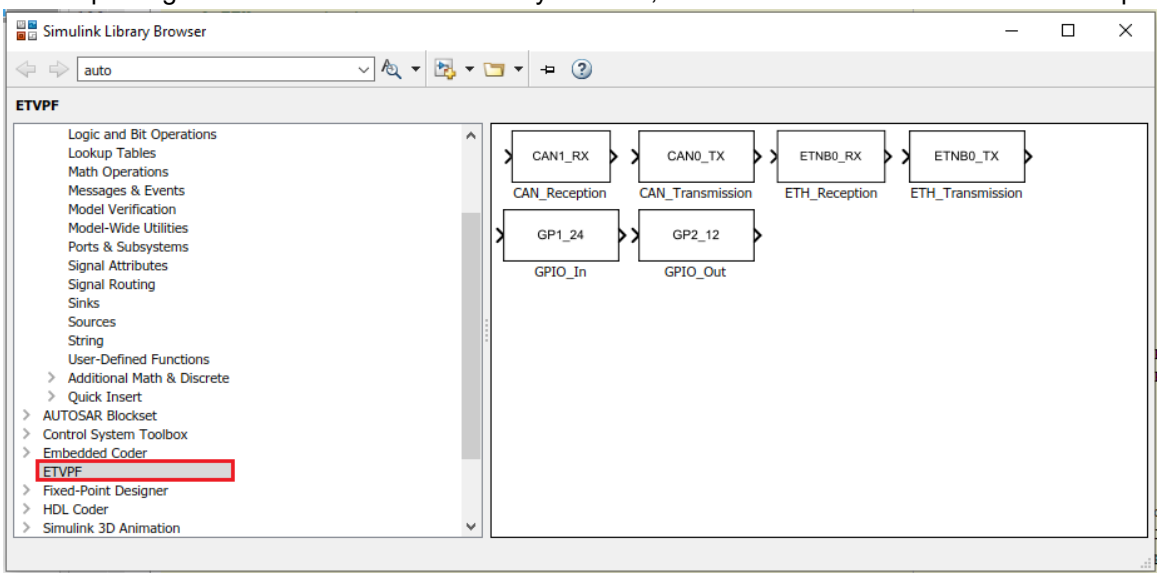


Figure 3-27 The S-function blocks of peripherals are in ETVPF package for R-Car S4

The sample model is Power Window model (slexPowerWindowExample.slx) that is used for the following explanation with ET-VPF. The feature of peripherals GPIO, RS-CANFD, ETHERNET will be supported by S-Function blocks that are added under the Code generation target.

Remarks 1. All blocks in the first layer under the Code generation target must be wrapped in a Subsystem.
2. The S-Function blocks of peripherals GPIO, RS-CANFD, ETHERNET must be in the layers under the Code generation target. If they are outside the Code generation target, the error can occur.

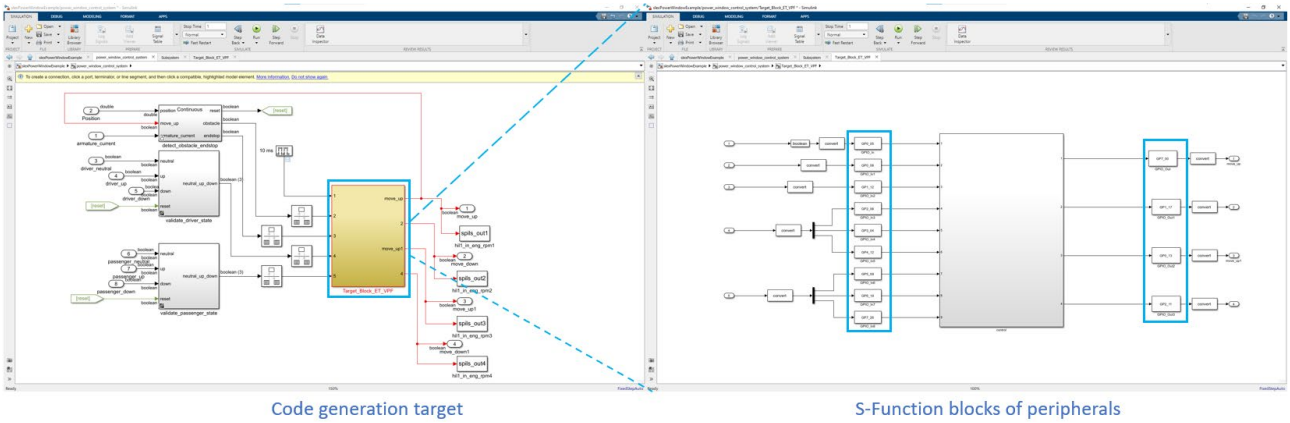


Figure 3-28 The Code generation target and S-Function blocks of peripherals for R-Car S4

The following tables show the information about the components of the sample model.

Table 3-5 Code generation target of sample model for R-Car S4

Sample model name	Code generation target	Block type
slexPowerWindowExample.slx	Target_Block_ET_VPF	Subsystem block

Table 3-6 Peripherals under Code generation target for R-Car S4

Code generation target	Peripheral	Peripheral port name	Block name	Block type
Target_Block_ET_VPF	GPIO	GP<Group>_<Pin>	GPIO_In<Number>	S-function block
			GPIO_Out<Number>	
	RS-CANFD	CAN<Unit>_TX	CAN_Transmission <Number>	
		CAN<Unit>_RX	CAN_Reception <Number>	
	ETHERNET	ETHA<Unit>_TX*1 ETNB<Unit>_TX*2	ETH_Transmission <Number>	
		ETHA<Unit>_RX*1 ETNB<Unit>_RX*2	ETH_Reception <Number>	

Remakes:

- *1: RSW2 function is supported by core CR52.
- *2: AVB-IF function is supported by core G4MH.

3.3.2 Setting configuration parameters

ET-VPF implements the execution of vHILS. environment generation by interworking with Embedded Coder. Therefore, it is necessary to check/set Embedded Coder options when using the vHILS environment generation functions provided by ET-VPF.

(1) Open MATLAB R2020a.
(2) Select [Current Folder] is a location that contains Power Window model. Open model, set model variables, and select port name for S-Function of peripherals.

(3) Open the [Model Configuration Parameters] dialog box to set for the Power Window model.

➢ **Step 1:** Select [Solver] -> [Type] is “Fixed-step”.

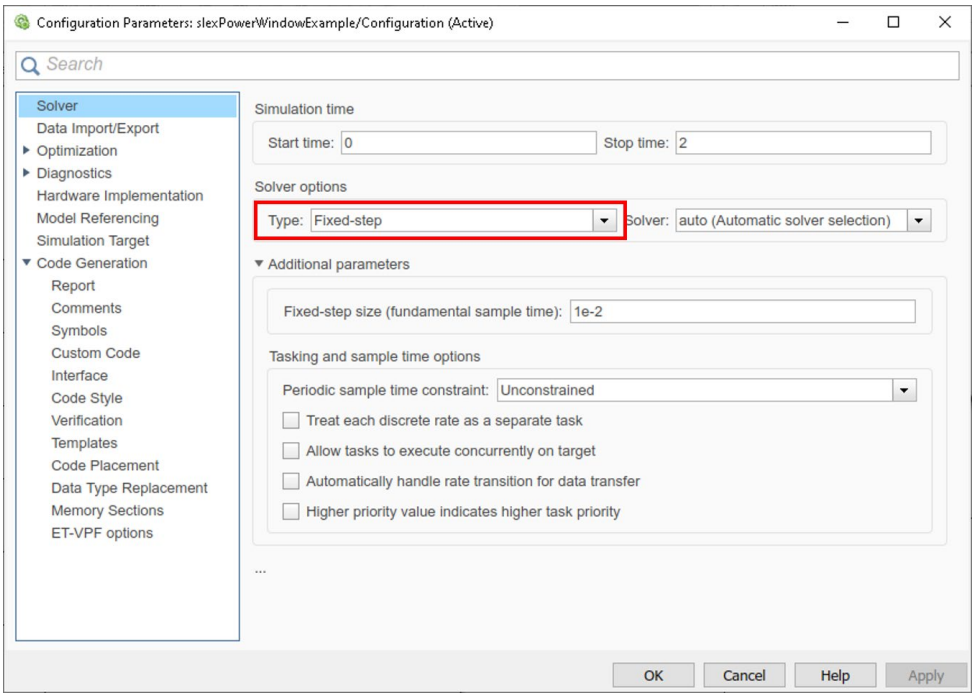


Figure 3-29 [Solver] settings for R-Car S4

➢ **Step 2:** Setting for [Code Generation].

- Select [System target file] is “etvpf.tlc”.
- Select [Generate code only].

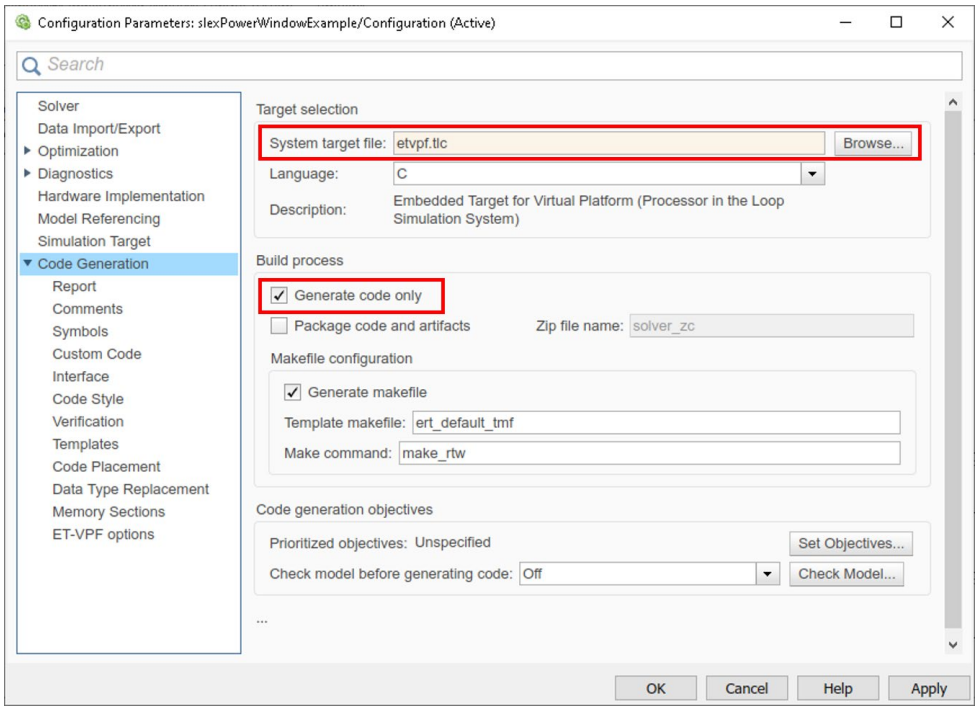


Figure 3-30 [Code Generation] settings for R-Car S4

➢ **Step 3:** In the [ET-VPF options], select necessary settings that described in the **Table 3-7 ET-VPF Options for R-Car S4.**

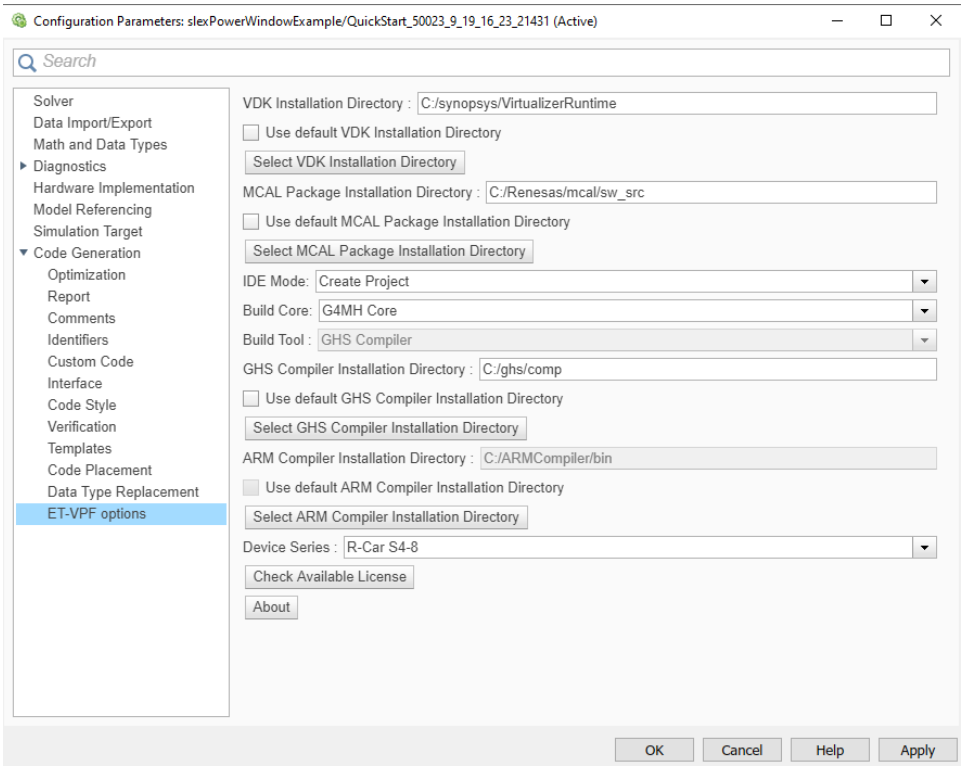


Figure 3-31 [ET-VPF options] settings for R-Car S4

The following table shows the items in the [ET-VPF options] pane.

Table 3-7 ET-VPF Options for R-Car S4

Item name	Description	
VDK Installation Directory *1 *14	Specifies the folder where VDK has been installed (the folder where SLS/windows/setup.bat file is stored) as an absolute path.	
[Use default VDK Installation Directory] checkbox	Specifies the default folder where VDK has been installed. It is “C:/synopsys/VirtualizerRuntime”.	
[Select VDK Installation Directory] button *1 *2	Clicking this button displays the dialog box for selecting the absolute path of the folder where the VDK is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [VDK Installation Directory] field.	
MCAL Package Installation Directory *3 *14	Specifies the folder where MCAL Package has been installed (the folder where g4mh_mcal/rel/S4/common_family/make/ghs /SampleApp.bat and cr52_mcal/rel/S4/common_family/make /arm/SampleApp.bat files are stored) as an absolute path.	
[Use default MCAL Package Installation Directory] checkbox	Specifies the default folder where MCAL Package has been installed. It is “C:/Renesas/mcal/sw_src”.	
[Select MCAL Package Installation Directory] button *3 *4	Clicking this button displays the dialog box for selecting the absolute path of the folder where the MCAL Package is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [MCAL Package Installation Directory] field.	
IDE Mode	Selects the type of project file that is loaded when the VDK starts and whether a series of processing including the download of a load module is performed after the VDK start-up.	
	Create Project (default)	The default project file provided by ET-VPF is loaded.
Build Core	Selects the core name of the microcontroller being used	
	G4MH Core (default)	This is default value of Build Core that will automatically be provided by ET-VPF is loaded. The G4MH Core will be used to load image file
	CR52 Core	The CR52 Core will be used to load image file
Build Tool *5	Selects the Build tool for the generated project, this indicates the compiler will be used to generate image file	
	GHS Compiler	This is the default value of the Build Tool that will be automatically loaded by ET-VPF. The GHS compiler will automatically be used to create image files when the G4MH core is selected.
	ARM Compiler	The ARM compiler will automatically be used to create the image file when the CR52 core is selected.
GHS Compiler Installation Directory *6 *12 *14	Specifies the folder where GHS Compiler has been installed (the folder where ccrh850.exe is stored) as an absolute path.	
[Use default GHS Compiler Installation Directory] checkbox *12	Specifies the default folder where GHS Compiler has been installed. It is “C:/ghs/comp”.	
[Select GHS Compiler Installation Directory] button *6 *7 *12	Clicking this button displays the dialog box for selecting the absolute path of the folder where the GHS Compiler is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [GHS Compiler Installation Directory] field.	
ARM Compiler Installation Directory *8 *13 *14	Specifies the folder where ARM Compiler has been installed (the folder where armclang.exe and armlink.exe are stored) as an absolute path.	
[Use default ARM Compiler Installation Directory] checkbox *13	Specifies the default folder where ARM Compiler has been installed. It is “C:/ARMCompiler/bin”.	

[Select ARM Compiler Installation Directory] button *8 *9 *13	Clicking this button displays the dialog box for selecting the absolute path of the folder where the ARM Compiler is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [ARM Compiler Installation Directory] field.	
Device Series *10	Selects the series name of the microcontroller being used.	
	<Device Series Name>	The supported Device Series described in **Table 1-1 Supported devices**.
	N/A	The default value of Device Series will automatically be selected when there is no license available.
[Check Available License] button *11	Displays list of available requiring licenses in ET-VPF System.	
[About] button	Displays version information and copyright information of ET-VPF.	

*1… When VDK has not been installed in the folder specified with the dialog box (SLS/windows/setup.bat file do not exist in the specified folder), an error is an output, and the information of the specified folder is not reflected in [VDK Installation Directory].
*2… When the [Use default VDK Installation Directory] checkbox is checked if the [Select VDK Installation Directory] button is clicked, an error message displays.
*3… When MCAL Package has not been installed in the folder specified with the dialog box (g4mh_mcal/rel/S4/common_family/make/ghs/SampleApp.bat and cr52_mcal/rel/S4/common_family/make/arm/SampleApp.bat files do not exist in the specified folder), an error is an output, and the information of the specified folder is not reflected in [MCAL Package Installation Directory].
*4… When the [Use default MCAL Package Installation Directory] checkbox is checked, if the [Select MCAL Package Installation Directory] button is clicked, an error message displays.
*5… This setting is only valid if “Create Project” is selected for [IDE Mode].
- When selecting “G4MH Core” for [Build Core], “GHS Compiler” is automatically selected for [Build Tools] and cannot be selected by the user.
- When selecting “CR52 Core” for [Build Core], “ARM Compiler” is automatically selected for [Build Tool] and cannot be selected by the user.

*6… When GHS Compiler has not been installed in the folder specified with the dialog box (ccrh850.exe file does not exist in the specified folder), an error is an output, and the information of the specified folder is not reflected in [GHS Compiler Installation Directory].
*7… When the [Use default GHS Compiler Installation Directory] checkbox is checked, if the [Select GHS Compiler Installation Directory] button is clicked, an error message displays.
*8… When ARM Compiler has not been installed in the folder specified with the dialog box (armclang.exe and armlink.exe files do not exist in the specified folder), an error is an output, and the information of the specified folder is not reflected in [ARM Compiler Installation Directory].
*9… When the [Use default ARM Compiler Installation Directory] checkbox is checked, if the [Select ARM Compiler Installation Directory] button is clicked, an error message displays.
*10… The list of supported devices in the current ET-VPF version (refer to **Table 1-1 Supported devices**).
*11… When clicking on this button, the dialog box displays and shows the list of required licenses. These can be used freely.
*12… This setting is only valid if [GHS Compiler] is selected for “Build Tool”.
*13… This setting is only valid if [ARM Compiler] is selected for “Build Tool”.
*14… The path of VDK, MCAL, GHS, ARM is only supporting the special characters that described in the following table.

Table 3-8 The supported special characters for R-Car S4

Special Characters	VDK	MCAL	GHS	ARM	<ET-VPF installation folder>
!	-	-	-	-	O
@	O	O	O	O	O
#	-	O	O	O	-
$	O	-	-	-	-
%	-	-	-	-	O
^	-	-	-	-	O
&	-	-	-	-	O
~	O	O	O	O	O
`	O	O	O	O	O
-	O	O	O	O	O
_	O	O	O	O	O
+	O	O	O	O	O
=	-	-	-	-	O
(	-	-	-	-	-
)	-	-	-	-	-
()	-	-	-	-	-
[	-	-	O	-	O
]	-	-	O	-	O
{	O	O	O	O	O
}	O	O	O	O	O
,	O	-	-	-	-
.	O	-	-	-	-
‘	O	-	-	-	-
‘’	-	-	-	-	-
;	-	-	-	-	-
space	-	-	-	-	-
*	-	-	-	-	-
/	-	-	-	-	-
|	-	-	-	-	-

* O: Can use
-: Cannot use

- **Step 4:** Click [Apply] button then save model.
- **Step 5:** Click [OK] or [X] button to close the [Model Configuration Parameters] dialog box.

3.3.3 Generating a vHILS environment

This section explains how to execute the generation of the vHILS environment required for vHILS.

ET-VPF provides the following command, which can be used in the MATLAB command window. This command automatically executes a series of operations for the generation of a vHILS environment.

Table 3-9 Provided command for R-Car S4

Command name	Description	Device
run_vdk	Generates a vHILS environment and executes vHILS automatically	R-Car S4

(4) Select the Code generation target on the model.
(5) Select target hardware processor type (Only do this step when the build core is CR52 Core).
- **Step 1:** Right click on the Code generation target block.
  - Select C/C++ Code.
  - Select Embedded Coder Quick Start.

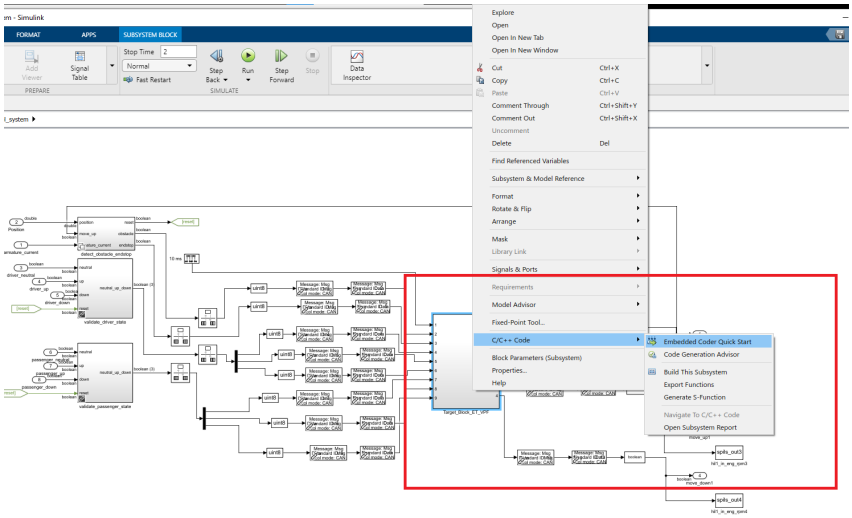


Figure 3-32 Open Embedded Coder Quick Start

➢ **Step 2:** Execution step by step to Word Size slide as the following figure.

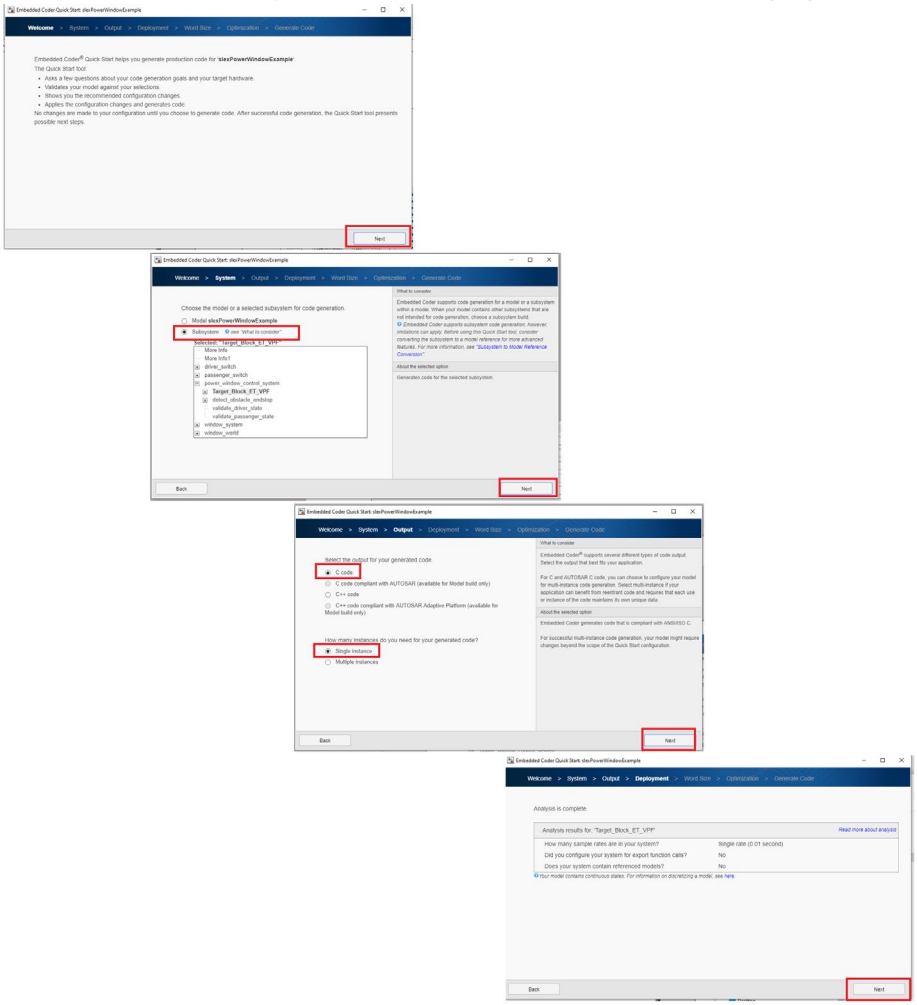


Figure 3-33 Switch to Word Size slide

➢ **Step 3:** Setting Device Vendor to ARM Compatible and Device Type to ARM Cortex as the following figure.

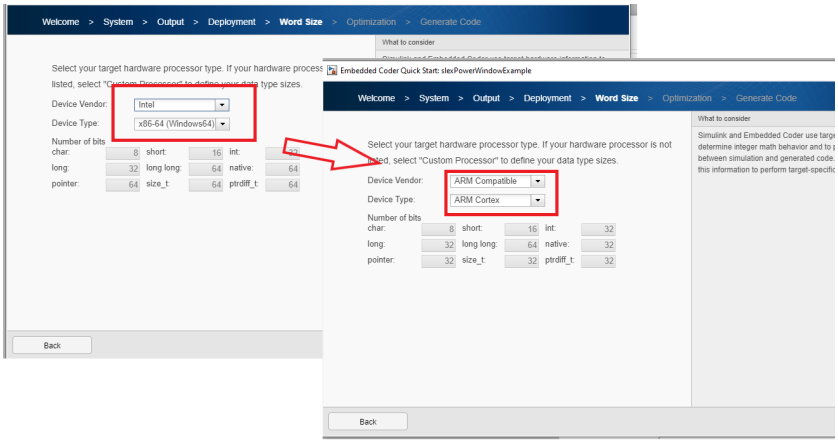


Figure 3-34 Setting Device Vendor and Device Type

➢ **Step 4:** Execution step by step to finish setting as the following figure.

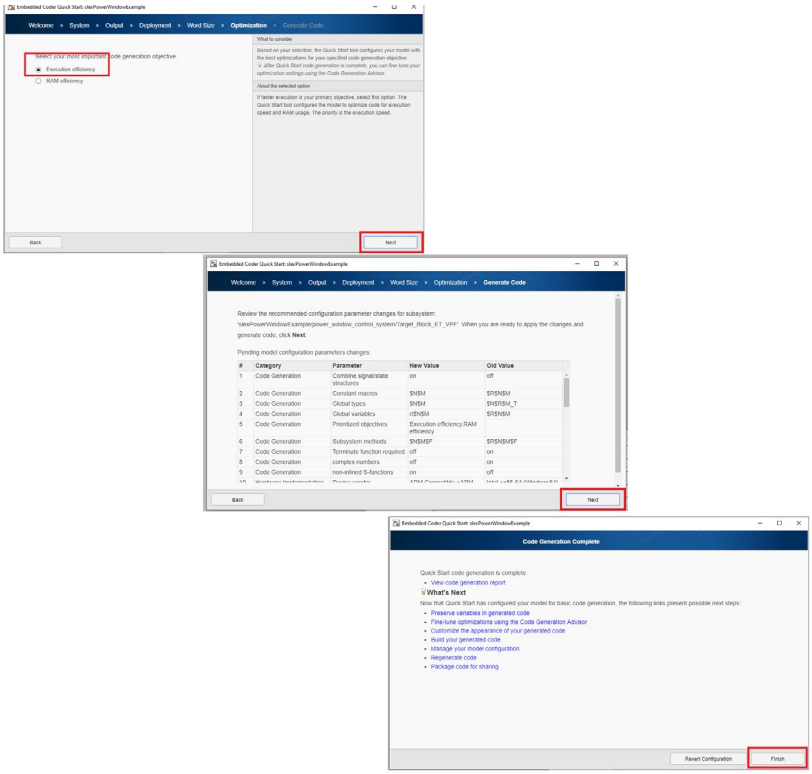


Figure 3-35 Setting Device Vendor and Device Type

➢ **Step 5:** Save model.

(6) Execute generation of vHILS environment by entering the provided command in the MATLAB Command Window, using the following syntax.
Here ">>" denotes the command prompt and "[Enter]" denotes the entry of the Enter key.

```
>> run_vdk [Enter]
```

3.3.3.1 Generating the target vHILS environment

(7) The target vHILS environment will be generated in the working directory (the location which contains the target model).
The target vHILS environment includes “slprj” and “<Code generation target>_etvpf” folders. And the necessary files will be generated in these folders.

- Source (<Code generation target>.c, <Code generation target>.h, <Code generation target>.mk, <Code generation target>_data.c, …).
- C source and header files (*.c,*.h,…).
- Peripherals’ wrapper source files (*.c)
- Image file: etvpf_main.out (G4MH core) or etvpf_main.elf (CR52 core)
- VP config file: *.vpcfg
- Python files: *.py
- Simulink Model file: *.slx
- Functional Mock-up Unit file: *.fmu
- Makefile: *.mak
- Batch file: *.bat

The model file is copied (the destination model file has the same name as the original model file but "_etvpf" suffix is added).
The Subsystem under the Code generation target block is replaced with the block for vHILS sequential execution (with block name is “VDK Bridge”) for the model file to be copied.

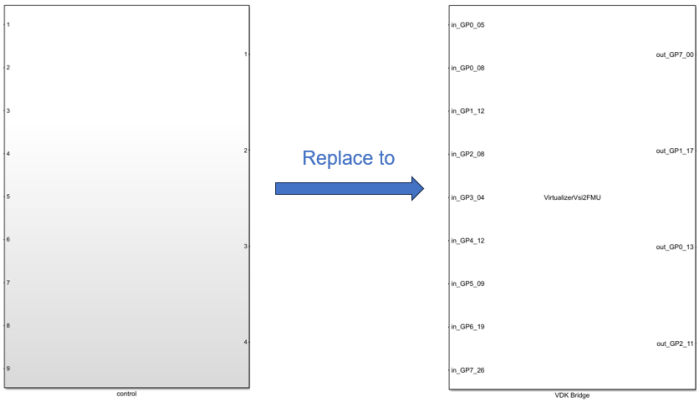


Figure 3-36 Example of replacing to the block for vHILS sequential execution for R-Car S4

3.3.4 Compiling generated source code

(8) After that, the C source files, and source files of the target device (with the extension is *.c, *. s, *.850) will be compiled to object files (with the extension is *.o, *.ao) via GNU Make.
The information related to the compilation will be displayed on MATLAB Command Window.

Figure 3-37 Compiling is displayed on MATLAB Command Window for R-Car S4

3.3.5 Executing vHILS

3.3.5.1 Create VDK project and import VP config

(9) After that, the VDK Window will be displayed. Users can configure detailed settings as below:

- **Step 1:** Select [File] -> [New] -> [VDK Project] to create a new VDK project.

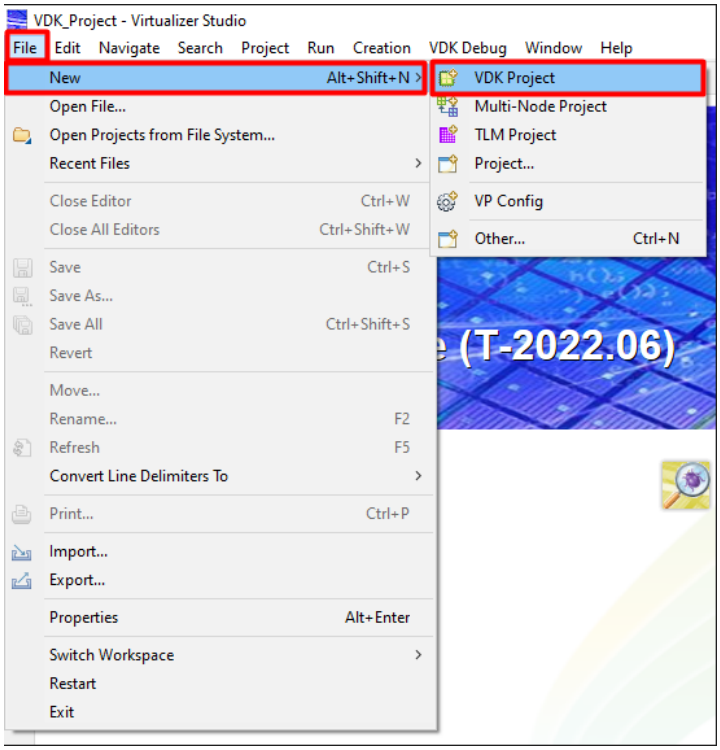


Figure 3-38 Creating new VDK project for R-Car S4

- **Step 2:** Create project name and project type.
  - Filling [Project Name] (1) of VDK project (**Note:** The [Use default location] (2) must be selected)
  - Select project type (Fixed VDK) in [Project type] (3) and then select template (RCar_S4_system (4.6)) in [Template] (4)
  - Click [Next] button (5) -> [Deselect All] button (6) -> [Finish] button (7) to complete this step.

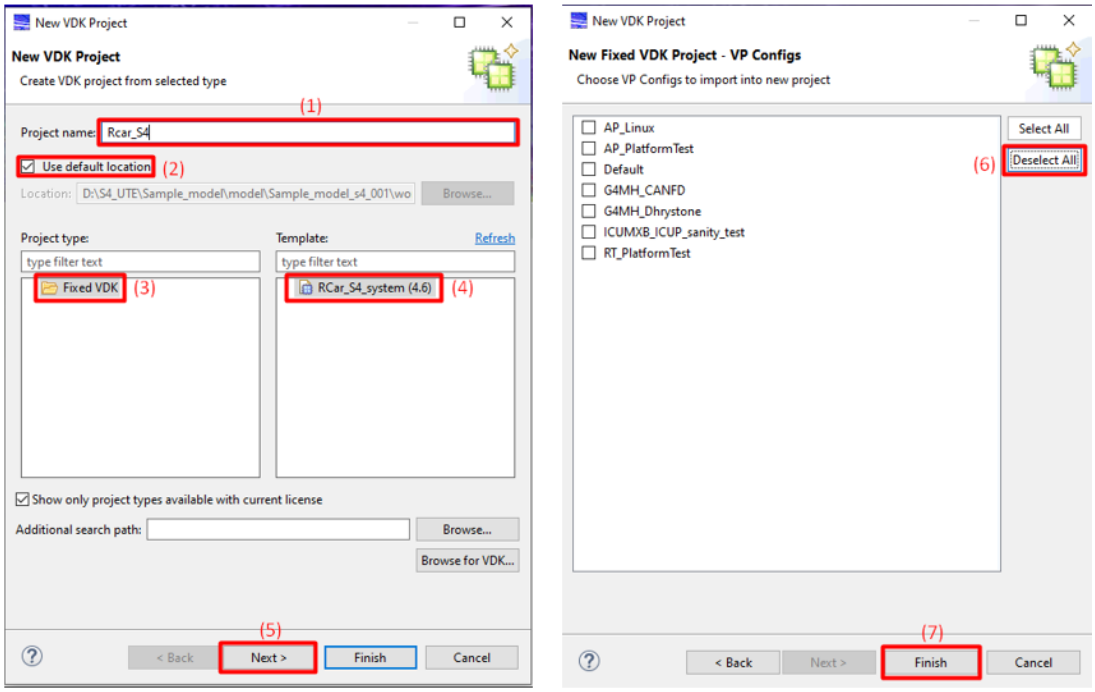


Figure 3-39 Creating project name and project type for R-Car S4

- **Step 3:** Click [Restore] to Show Project Explorer

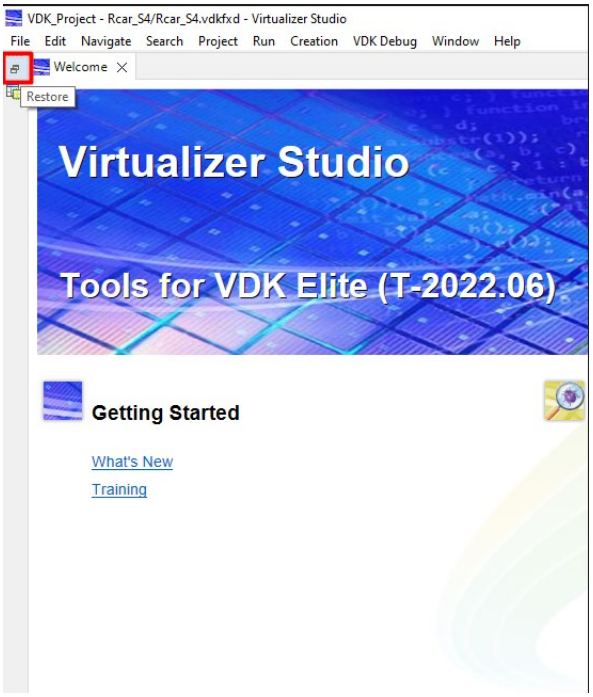


Figure 3-40 Show Project Explorer for R-Car S4

- **Step 4:** Import available VP config file (this file will be generated in the working directory)
  - In [Project Explorer], click right mouse [vpconfigs] (1) and then select [Import] (2) to select the VP config.
  - When Import Window is opened, select Import VP Config (3) in [Virtual Prototyping] and then click [Next] button (4).
  - Select the full path of VP config file (5): Select “S4_G4MH_VP_Config.vpcfg” file if “Build Core” is selected for [G4MH Core] else select “S4_CR52_VP_Config.vpcfg” file if “Build Core” is selected for [CR52 Core]
  - Click [Finish] button (6) to complete the import VP config file.

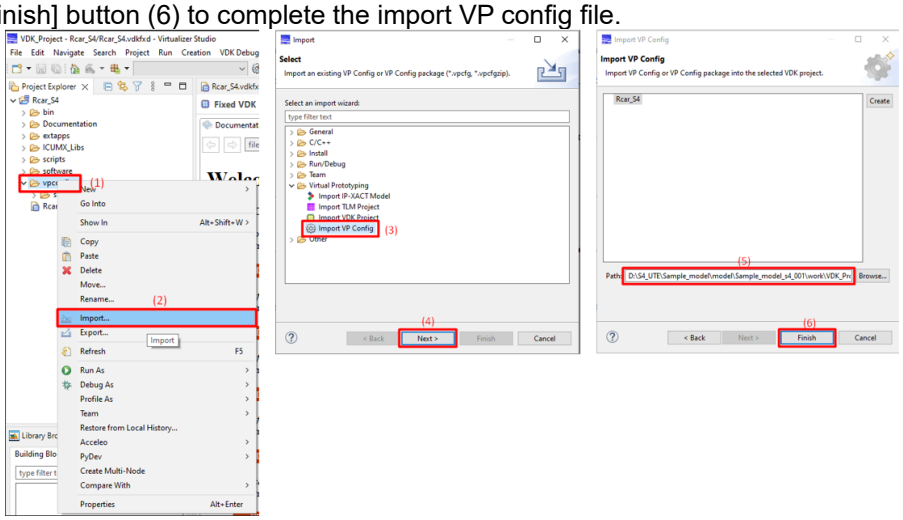


Figure 3-41 Importing available VP config file for R-Car S4

- **Step 5:** Run simulation: Click [Run] button to start simulation.

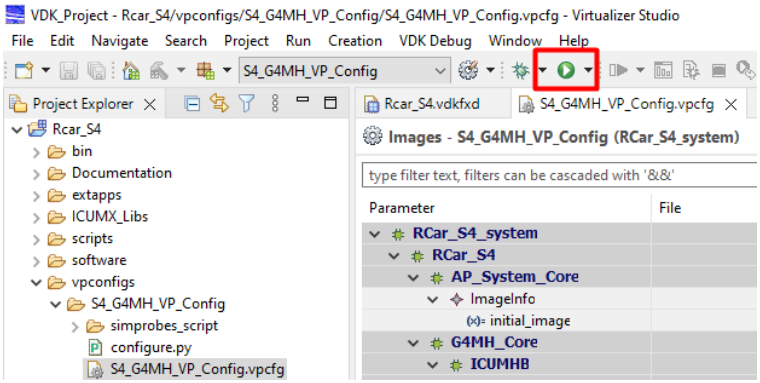


Figure 3-42 Run simulation for R-Car S4

- **Step 6:** Continue simulation: After the starting simulation is completed successfully, click [Resume suspended simulation] button to continue the simulation.

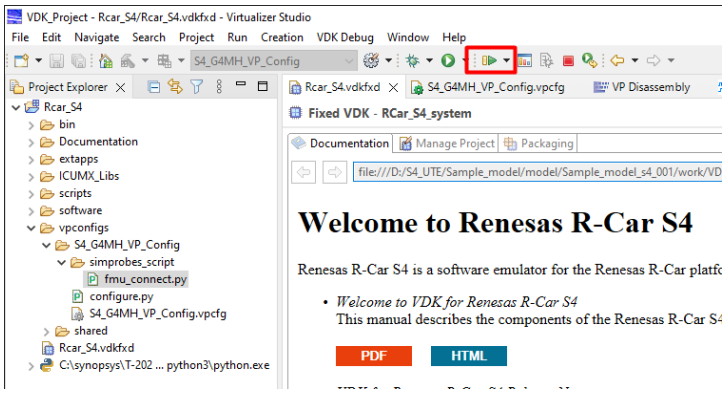


Figure 3-43 Continue simulation for R-Car S4

3.3.5.2 Result of executing vHILS

(10) The vHILS execution starts on both the Simulink model and VDK.
Transmission and reception protocol between MATLAB and VDK is periodically after sampling time.

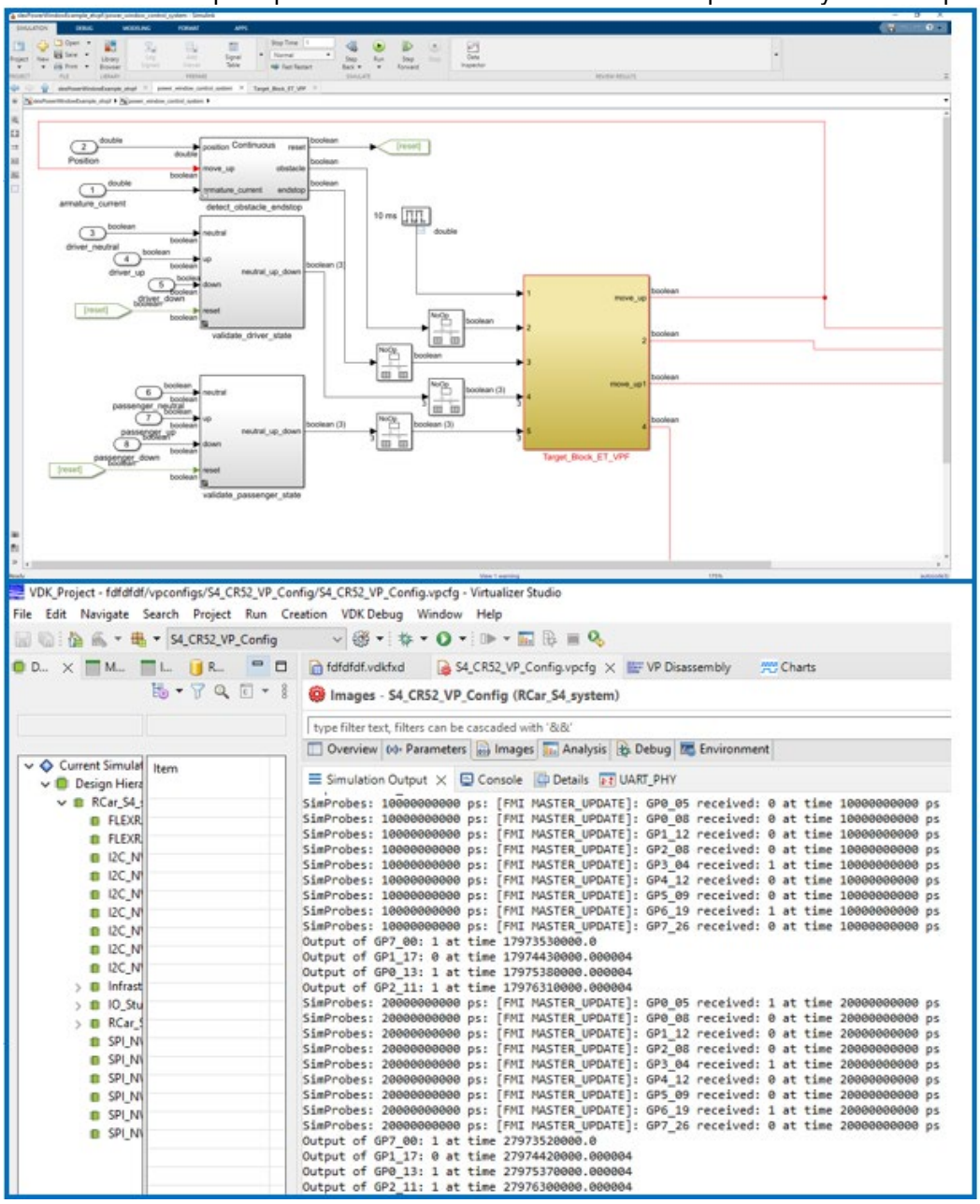


Figure 3-44 The vHILS execution for S4

(11) The result of scope is below.

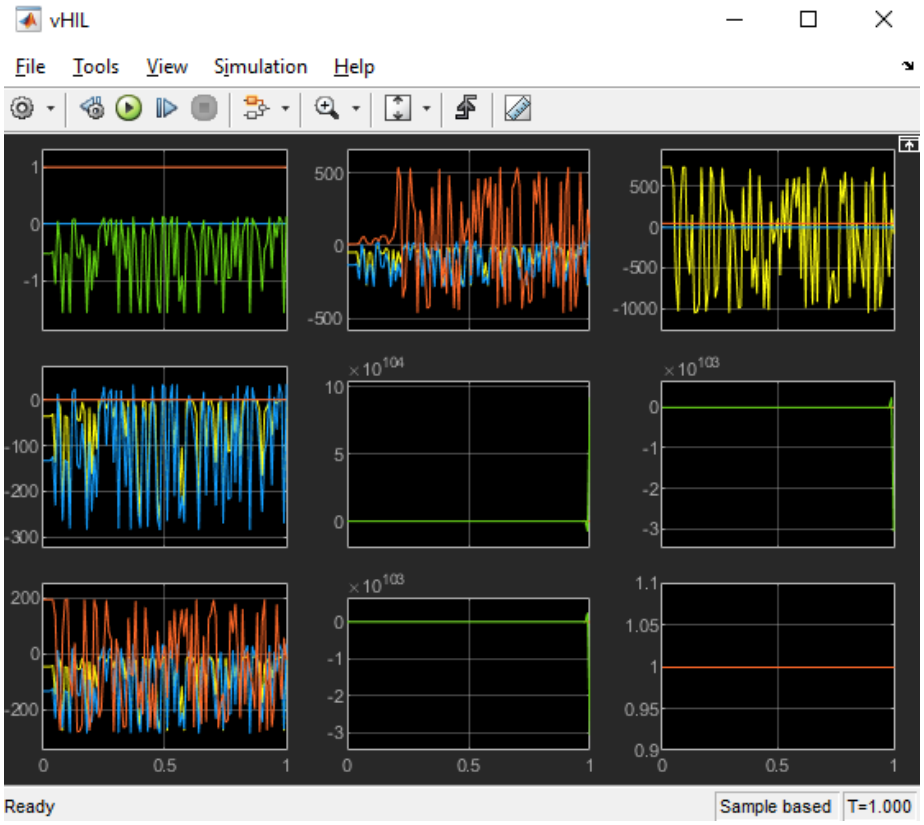


Figure 3-45 The result of scope for R-Car S4

3.4 Executing Virtual Hardware in the Loop Simulation R-Car V4H

The following describes how to generate a vHILS environment necessary for Simulator Processor in the Loop Simulation (hereafter referred to as vHILS) by ET-VPF.

3.4.1 Embedded sample model

Firstly, the user needs to open ETVPF package of the MATLAB Simulink Library Browser to choose the expected S-function blocks of peripherals.

- The ETVPF package is in MATLAB Simulink Library Browser, which contains the S-function blocks of peripherals.

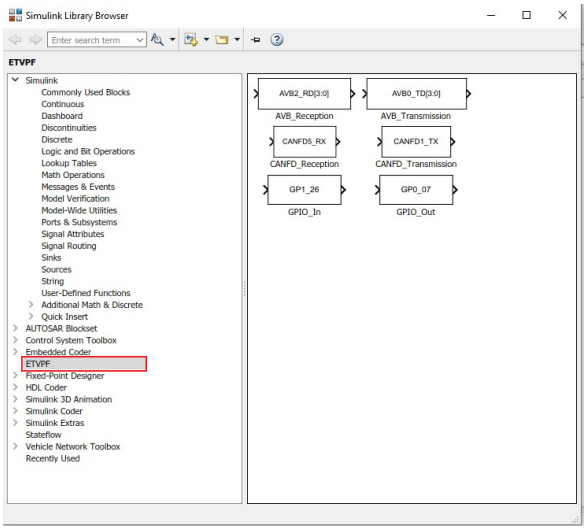


Figure 3-46 The S-function blocks of peripherals are in ETVPF package for R-Car V4H

The sample model is Power Window model (slexPowerWindowExample.slx) that is used for the following explanation with ET-VPF. The feature of peripherals GPIO, RS-CANFD, ETHERNET will be supported by S-Function blocks that are added under the Code generation target.

Remarks
1. All blocks in the first layer under the Code generation target must be wrapped in a Subsystem.
2. The S-Function blocks of peripherals GPIO, RS-CANFD, ETHERNET must be in the layers under the Code generation target. If they are outside the Code generation target, the error can occur.

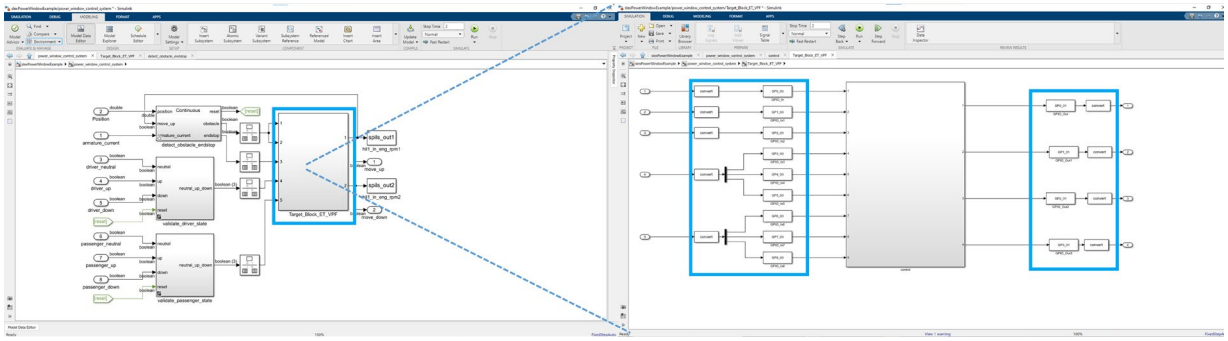


Figure 3-47 The Code generation target and S-Function blocks of peripherals for R-Car V4H

The following tables show the information about the components of the sample model.

Table 3-10 Code generation target of sample model for R-Car V4H

Sample model name	Code generation target	Block type
slexPowerWindowExample.slx	Target_Block_ET_VPF	Subsystem block

Table 3-11 Peripherals under Code generation target for R-Car V4H

Code generation target	Peripheral	Peripheral port name	Block name	Block type
Target_Block_ET_VPF	GPIO	GP<Group>_<Pin>	GPIO_In<Number>	S-function block
			GPIO_Out<Number>	
	RS-CANFD	CANFD<channel>_TX	CAN_Transmission<Number>	S-function block
		CANFD<channel>_RX	CAN_Reception<Number>	
	ETHERNET	AVB<unit>_TD[3:0]	AVB_Transmission<Number>	S-function block
		AVB<unit>_RD[3:0]	AVB_Reception<Number>	S-function block

3.4.2 Setting configuration parameters

ET-VPF implements the execution of vHILS environment generation by interworking with Embedded Coder. Therefore, it is necessary to check/set Embedded Coder options when using the vHILS environment generation functions provided by ET-VPF.

(1) Open MATLAB R2020a.
(2) Select [Current Folder] is a location that contains Power Window model. Open model, set model variables, and select port name for S-Function of peripherals.
(3) Open the [Model Configuration Parameters] dialog box to set for the Power Window model.
➢ **Step 1:** Select [Solver] -> [Type] is “Fixed-step”.

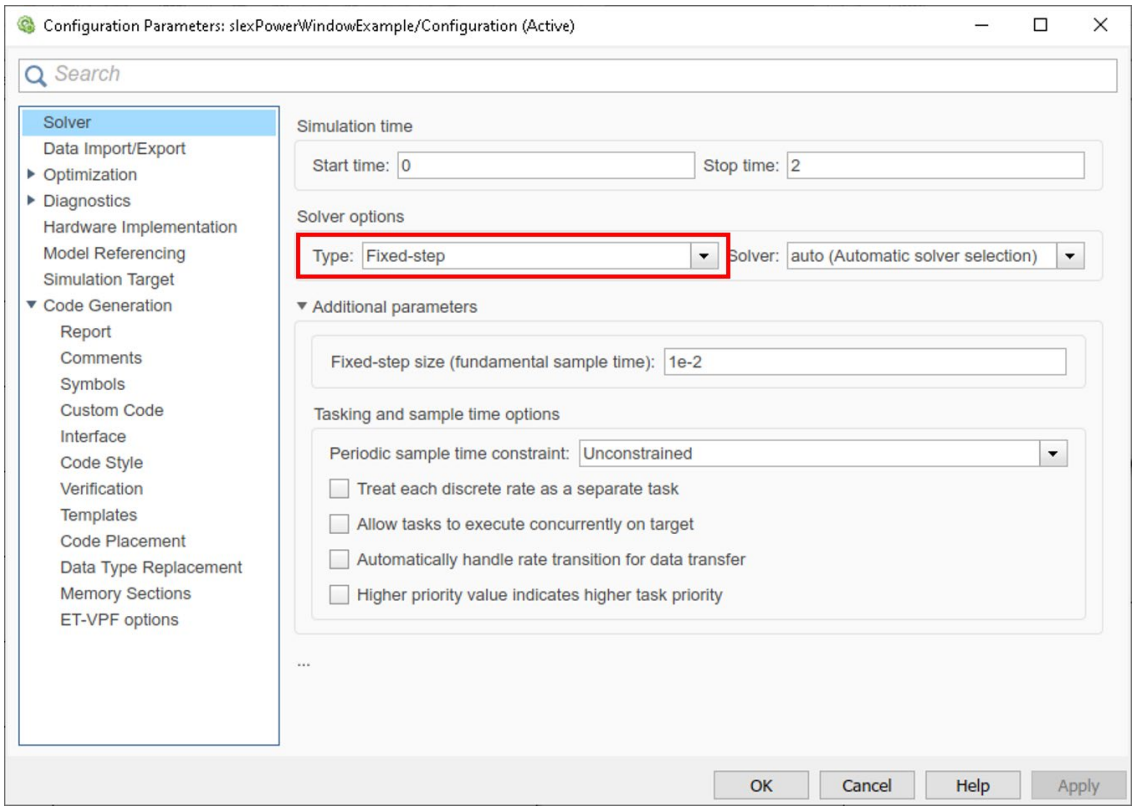


Figure 3-48 [Solver] settings for R-Car V4H

- **Step 2:** Setting for [Code Generation].
  - Select [System target file] is “etvpf.tlc”.
  - Select [Generate code only].

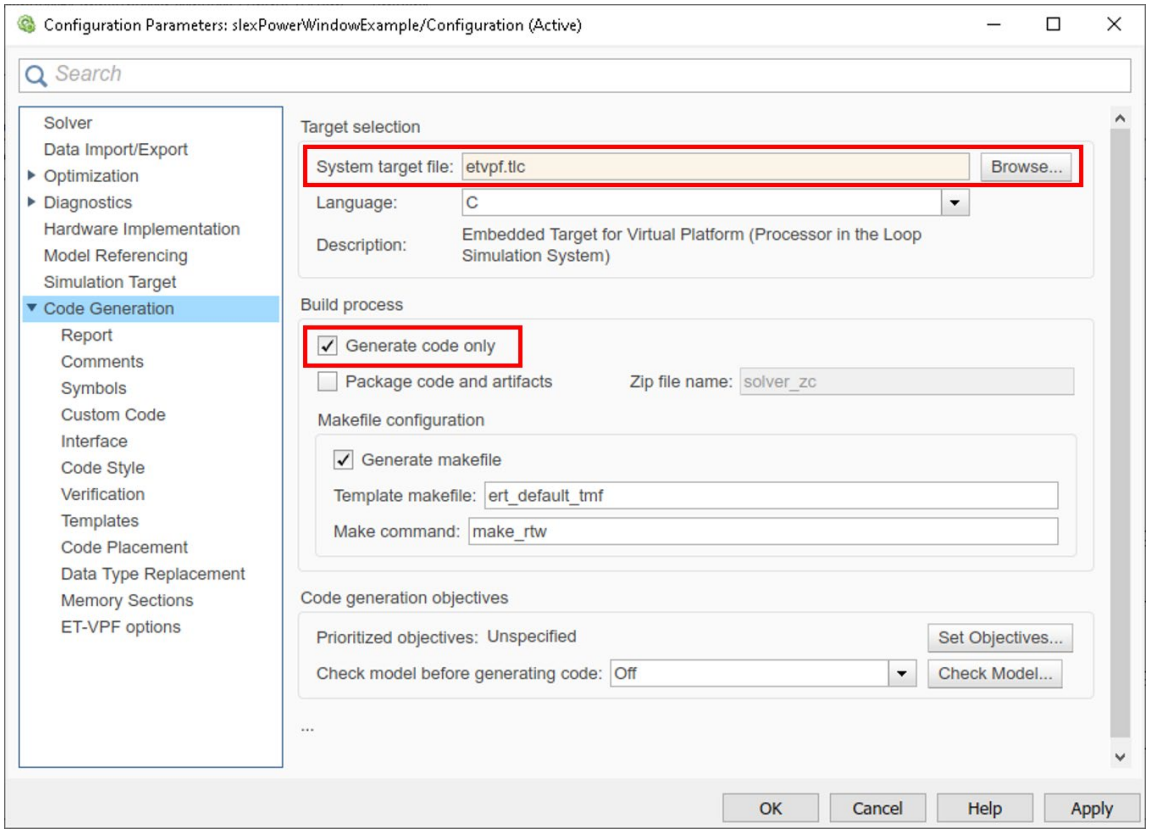


Figure 3-49 [Code Generation] settings for R-Car V4H

Step 3: In the [ET-VPF options], select necessary settings that described in the **Table 3-12 ET-VPF Options for R-Car V4H.**

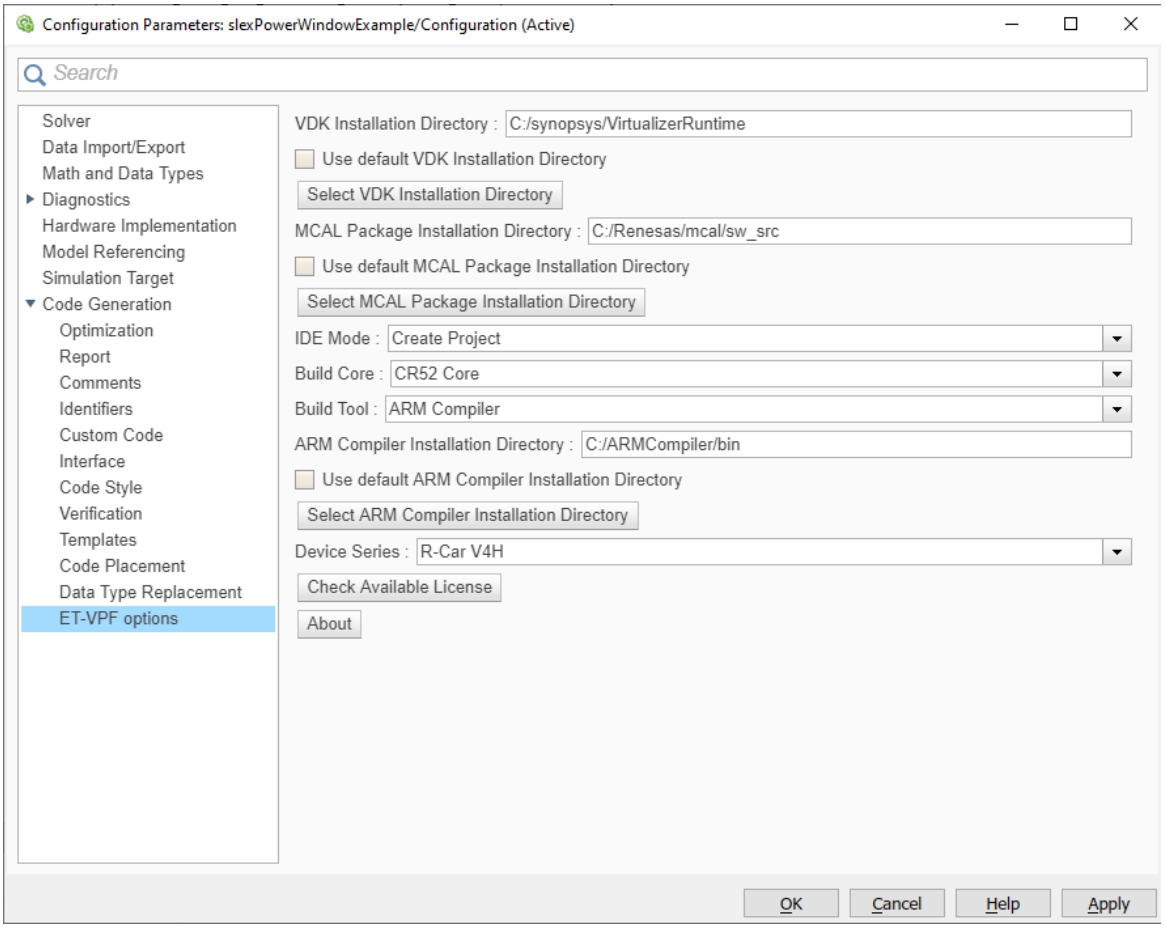


Figure 3-50 [ET-VPF options] settings for R-Car V4H

The following table shows the items in the [ET-VPF options] pane.

Table 3-12 ET-VPF Options for R-Car V4H

Item name	Description	
VDK Installation Directory *1 *10	Specifies the folder where VDK has been installed (the folder where SLS/windows/setup.bat file is stored) as an absolute path.	
[Use default VDK Installation Directory] checkbox	Specifies the default folder where VDK has been installed. It is “C:/synopsys/VirtualizerRuntime”.	
[Select VDK Installation Directory] button *1 *2	Clicking this button displays the dialog box for selecting the absolute path of the folder where the VDK is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [VDK Installation Directory] field.	
MCAL Package Installation Directory *3 *10	Specifies the folder where MCAL Package has been installed as an absolute path.	
[Use default MCAL Package Installation Directory] checkbox	Specifies the default folder where MCAL Package has been installed. It is “C:/Renesas/mcal/sw_src”.	
[Select MCAL Package Installation Directory] button *3 *4	Clicking this button displays the dialog box for selecting the absolute path of the folder where the MCAL Package is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [MCAL Package Installation Directory] field.	
IDE Mode	Selects the type of project file that is loaded when the VDK starts and whether a series of processing including the download of a load module is performed after the VDK start-up.	
	Create Project (default)	The default project file provided by ET-VPF is loaded.
Build Core	Selects the core name of the microcontroller being used	
	CR52 Core	The default value of Build Core will automatically be provided by ET-VPF is loaded
Build Tool	Selects the Build tool for the generated project, this indicates the compiler will be used to generate image file	
	ARM Compiler	This is default value of Build Tool that will automatically be provided by ET-VPF is loaded. The ARM Compiler will be used to generate image file
ARM Compiler Installation Directory *5 * 9 *10	Specifies the folder where ARM Compiler has been installed (the folder where armclang.exe and armlink.exe are stored) as an absolute path.	
[Use default ARM Compiler Installation Directory] checkbox *9	Specifies the default folder where ARM Compiler has been installed. It is “C:/ARMCompiler/bin”.	
[Select ARM Compiler Installation Directory] button *5 *6 *9	Clicking this button displays the dialog box for selecting the absolute path of the folder where the ARM Compiler is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [ARM Compiler Installation Directory] field.	
Device Series *7	Selects the series name of the microcontroller being used.	
	<Device Series Name>	The supported Device Series described in **Table 1-1 Supported devices**.
	N/A	The default value of Device Series will automatically be selected when there is no license available.
[Check Available License] button *8	Displays list of available requiring licenses in ET-VPF System.	
[About] button	Displays version information and copyright information of ET-VPF.	

*1… When VDK has not been installed in the folder specified with the dialog box (SLS/windows/setup.bat

file do not exist in the specified folder), an error is an output, and the information of the specified folder is not reflected in [VDK Installation Directory].
*2… When the [Use default VDK Installation Directory] checkbox is checked if the [Select VDK Installation Directory] button is clicked, an error message displays.
*3… When MCAL Package has not been installed in the folder specified with the dialog box (rel/V4H/common_family/make/arm/SampleApp.bat file do not exist in the specified folder), an error is an output, and the information of the specified folder is not reflected in [MCAL Package Installation Directory].
*4… When the [Use default MCAL Package Installation Directory] checkbox is checked, if the [Select MCAL Package Installation Directory] button is clicked, an error message displays.
*5… When ARM Compiler has not been installed in the folder specified with the dialog box (armclang.exe and armlink.exe files do not exist in the specified folder), an error is an output, and the information of the specified folder is not reflected in [ARM Compiler Installation Directory].
*6… When the [Use default ARM Compiler Installation Directory] checkbox is checked, if the [Select ARM Compiler Installation Directory] button is clicked, an error message displays.
*7… The list of supported devices in the current ET-VPF version (refer to **Table 1-1 Supported devices**).
*8… When clicking on this button, the dialog box displays and shows the list of required licenses. These can be used freely.
*9… This setting is only valid if “Build Core” is selected for [CR52 Core].
*10… The path of VDK, MCAL, ARM is only supporting the special characters that described in the following table.

Table 3-13 The supported special characters for R-Car V4H

Special Characters	VDK	MCAL	ARM	<ET-VPF installation folder>
!	-	-	-	O
@	O	O	O	O
#	-	O	O	-
$	O	-	-	-
%	-	-	-	O
^	-	-	-	O
&	-	-	-	O
~	O	O	O	O
`	O	O	O	O
-	O	O	O	O
_	O	O	O	O
+	O	O	O	O
=	-	-	-	O
(	-	-	-	-
)	-	-	-	-
()	-	-	-	-
[	-	-	-	O
]	-	-	-	O
{	O	O	O	O
}	O	O	O	O
,	O	-	-	-
.	O	-	-	-
‘	O	-	-	-
‘’	-	-	-	-
;	-	-	-	-
space	-	-	-	-
*	-	-	-	-
/	-	-	-	-
|	-	-	-	-

* O: Can use
-: Cannot use

Remark: Special characters for VDK, MCAL will update after coding phase for R-Car V4H

- **Step 4:** Click [Apply] button then save model.
- **Step 5:** Click [OK] or [X] button to close the [Model Configuration Parameters] dialog box.

3.4.3 Generating a vHILS environment

This section explains how to execute the generation of the vHILS environment required for vHILS.

ET-VPF provides the following command, which can be used in the MATLAB command window. This command automatically executes a series of operations for the generation of a vHILS environment.

Table 3-14 Provided command for R-Car V4H

Command name	Description	Device
run_vdk	Generates a vHILS environment and executes vHILS automatically	R-Car V4H

(4) Select the Code generation target on the model.
(5) Select target hardware processor type (Only do this step when the build core is CR52 Core).
- **Step 1:** Right click on the Code generation target block.
  - Select C/C++ Code.
  - Select Embedded Coder Quick Start.

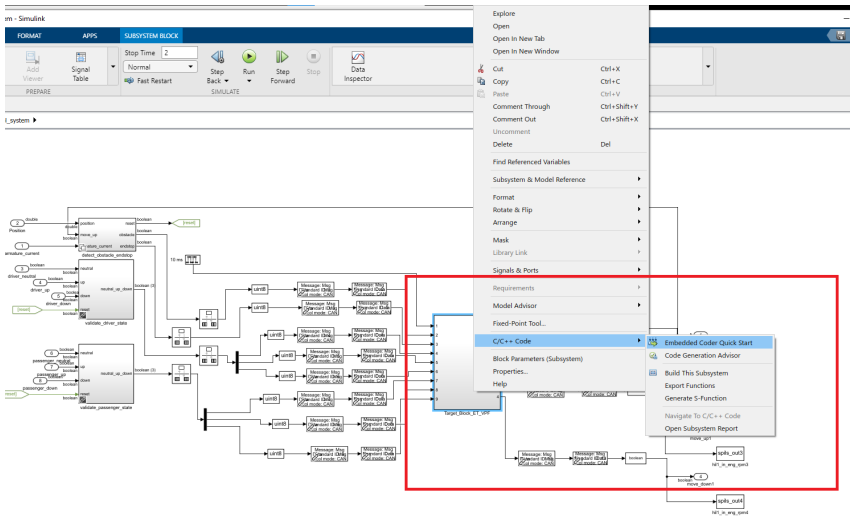


Figure 3-51 Open Embedded Coder Quick Start

➢ **Step 2:** Execution step by step to Word Size slide as the following figure.

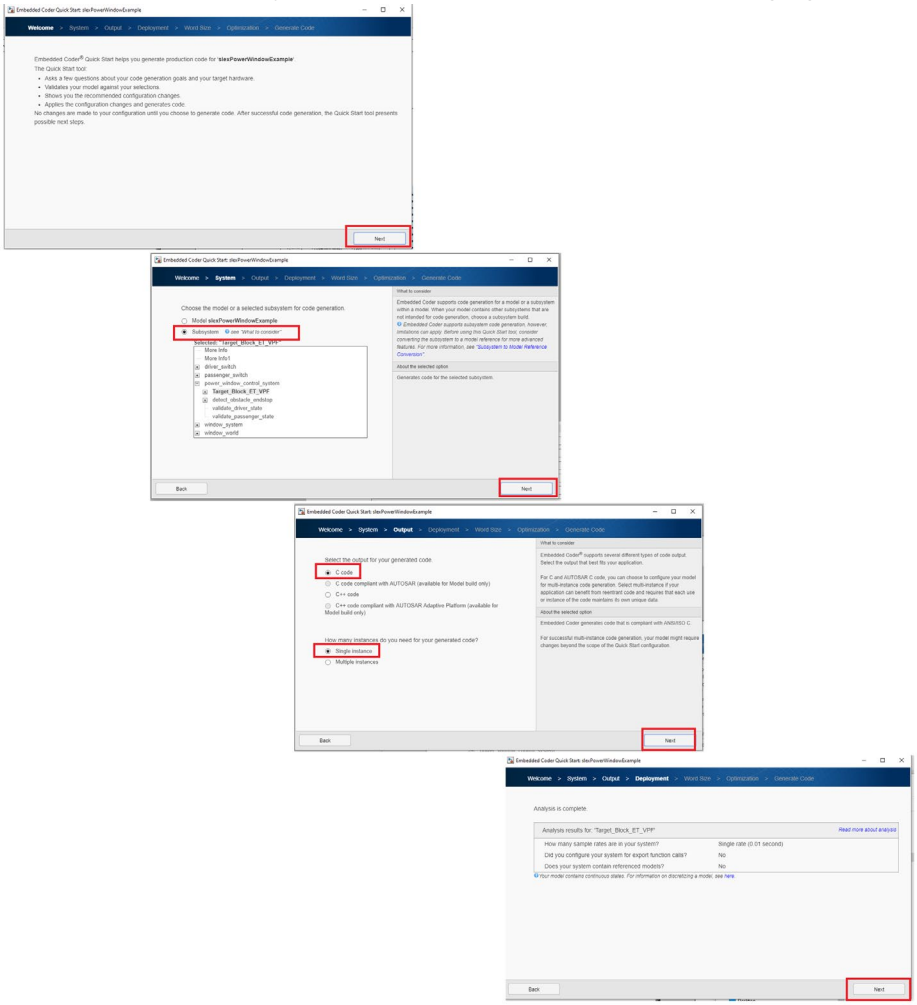


Figure 3-52 Switch to Word Size slide

➢ **Step 3:** Setting Device Vendor to ARM Compatible and Device Type to ARM Cortex as the following figure.

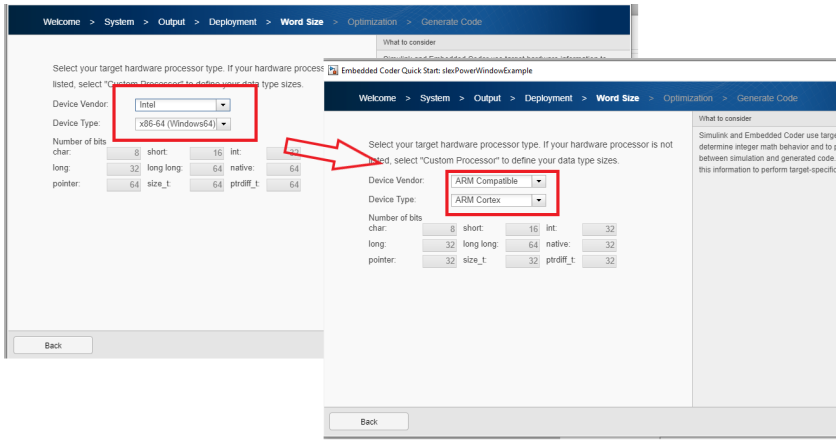


Figure 3-53 Setting Device Vendor and Device Type

➢ **Step 4:** Execution step by step to finish setting as the following figure.

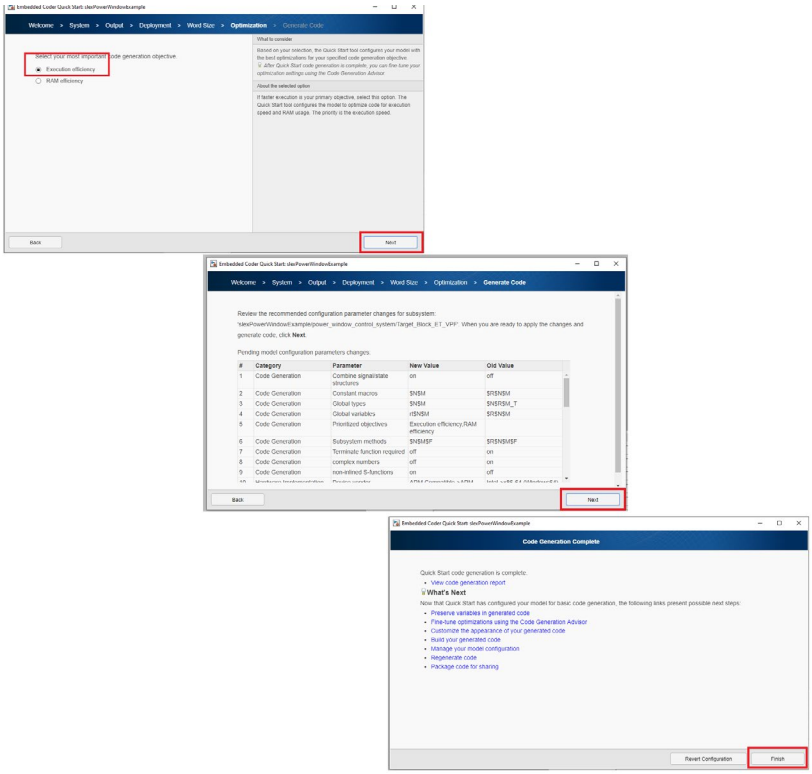


Figure 3-54 Setting Device Vendor and Device Type

➢ **Step 5:** Save model.

(6) Execute generation of vHILS environment by entering the provided command in the MATLAB Command Window, using the following syntax.
Here ">>" denotes the command prompt and "[Enter]" denotes the entry of the Enter key.

```
>> run_vdk [Enter]
```

3.4.3.1 Generating the target vHILS environment

(7) The target vHILS environment will be generated in the working directory (the location which contains the target model).
The target vHILS environment includes “slprj” and “<Code generation target>_etvpf” folders. And the necessary files will be generated in these folders.

- Source (<Code generation target>.c, <Code generation target>.h, <Code generation target>.mk, <Code generation target>_data.c, …).
- C source and header files (*.c,*.h,…).
- Peripherals’ wrapper source files (*.c)
- Image file: etvpf_main.out (G4MH core) or etvpf_main.elf (CR52 core)
- VP config file: *.vpcfg
- Python files: *.py
- Simulink Model file: *.slx
- Functional Mock-up Unit file: *.fmu
- Makefile: *.mak
- Batch file: *.bat

The model file is copied (the destination model file has the same name as the original model file but "_etvpf" suffix is added).
The Subsystem under the Code generation target block is replaced with the block for vHILS sequential execution (with block name is “VDK Bridge”) for the model file to be copied.

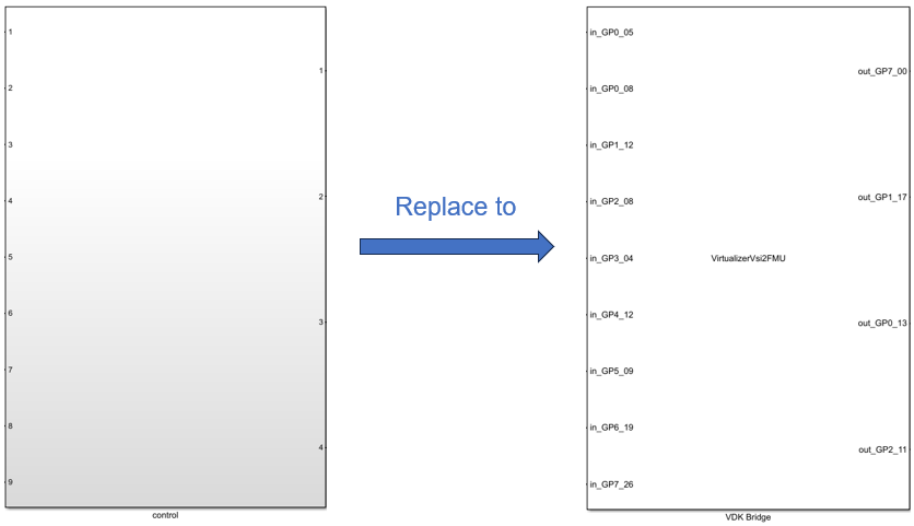


Figure 3-55 Example of replacing to the block for vHILS sequential execution for R-Car V4H

3.4.4 Compiling generated source code

(8) After that, the C source files, and source files of the target device (with the extension is *.c, *. s) will be compiled to object files (with the extension is *. o) via GNU Make.
The information related to the compiling will be displayed on MATLAB Command Window.

```
------------------------------------------------------------------------
[INCLUDES]
D:\Model\work\Renesas\make\arm\startup
------------------------------------------------------------------------
[AAA]  D:\Model\work\Renesas\make\..\Common_files_V4H\linker\App_Sample.scat
[FLAGS]      -c -O3 --target=arm-arm-none-eabi -g -gdwarf-2 -fno-short-enums -mfloat-abi=hard -nostdlib
[ASSEMBLING]...
[ASSEMBLY]... Done
------------------------------------------------------------------------
[COMPILER]   "C:\Workspace\ARMCompiler6.16.1\bin\armclang.exe"
[FLAGS]      -c -O3 --target=arm-arm-none-eabi -g -gdwarf-2 -fno-short-enums -mfloat-abi=hard -nostdlib
[COMPILING]...
  U:\rel\modules\port\src\Port.c U:\rel\modules\port\src\PFC\Port_PFC_LLDriver.c U:\rel\modules\port\sr
DD
```

Figure 3-56 Compiling is displayed on MATLAB Command Window for R-Car V4H

3.4.5 Executing vHILS

3.4.5.1 Create VDK project and import VP config

(9) After that, the VDK Window will be displayed. Users can configure detailed settings as below:

- **Step 1:** Select [File] -> [New] -> [VDK Project] to create new VDK project.

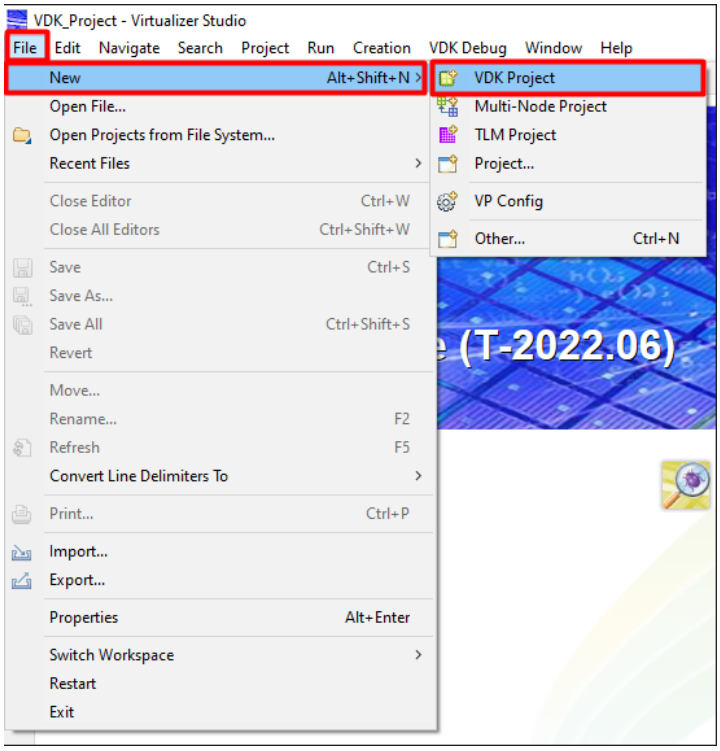


Figure 3-57 Creating new VDK project for R-Car V4H

- **Step 2:** Create project name and project type.
  - Filling [Project Name] (1) of VDK project (**Note:** The [Use default location] (2) must be selected)
  - Select project type (Fixed VDK) in [Project type] (3) and then select template (RCar_S4_system (4.6)) in [Template] (4)
  - Click [Next] button (5) -> [Deselect All] button (6) -> [Finish] button (7) to complete this step

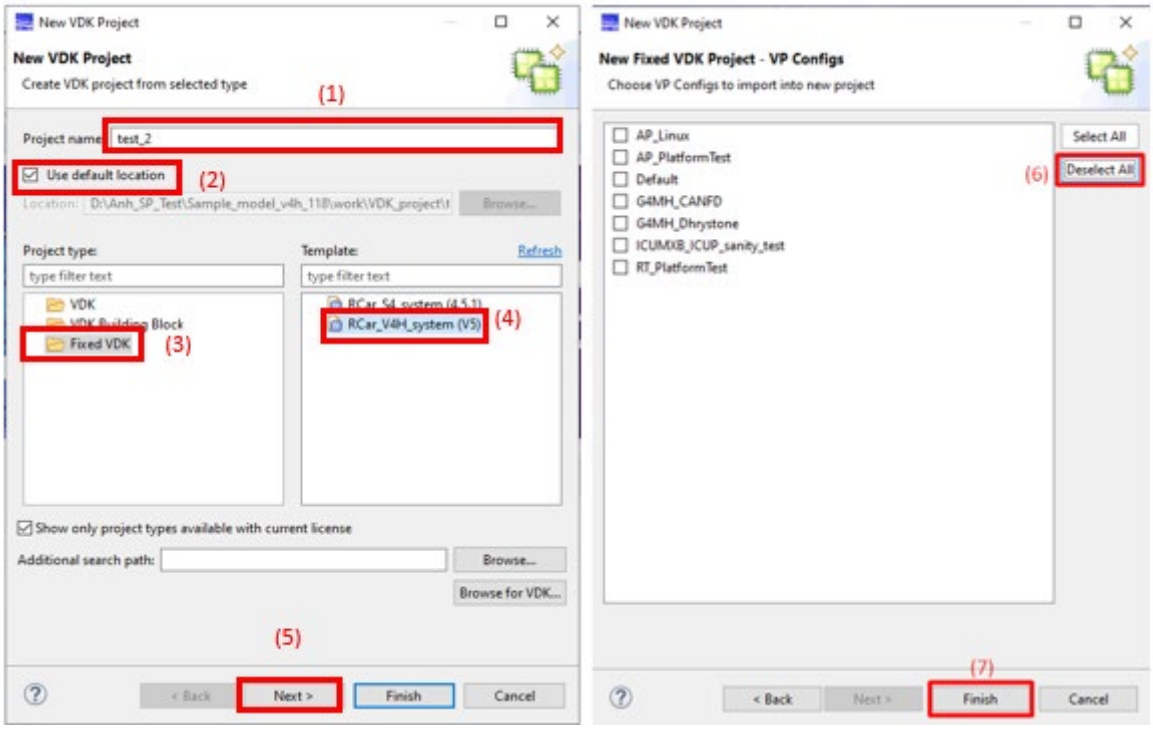


Figure 3-58 Creating project name and project type for R-Car V4H

➢ **Step 3:** Click [Restore] to Show Project Explorer

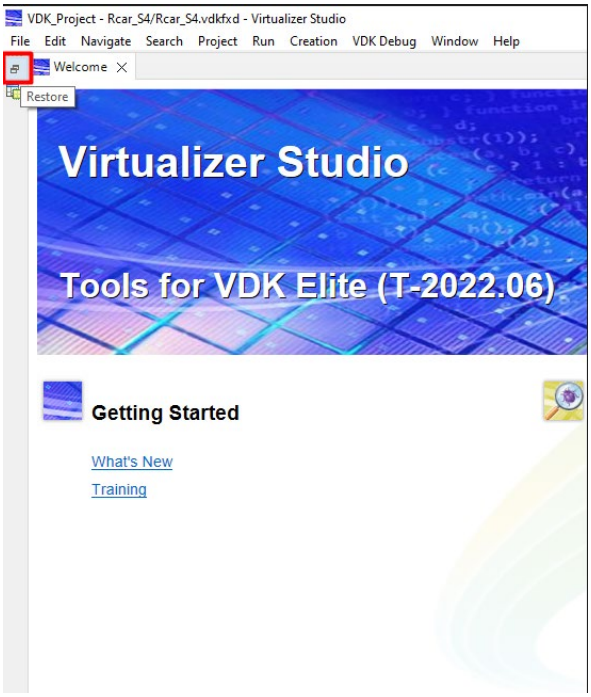


Figure 3-59 Creating project name and project type for R-Car V4H

➢ **Step 4:** Import available VP config file (this file will be generated in the working directory)

- In [Project Explorer], click right mouse [vpconfigs] (1) and then select [Import] (2) to select the VP config.
- When Import Window is opened, select Import VP Config (3) in [Virtual Prototyping] and then click [Next] button (4).
- Select the full path of VP config file (5): Select “V4H_DIO_FMU.vpcfg” file
- Click [Finish] button (6) to complete the import VP config file

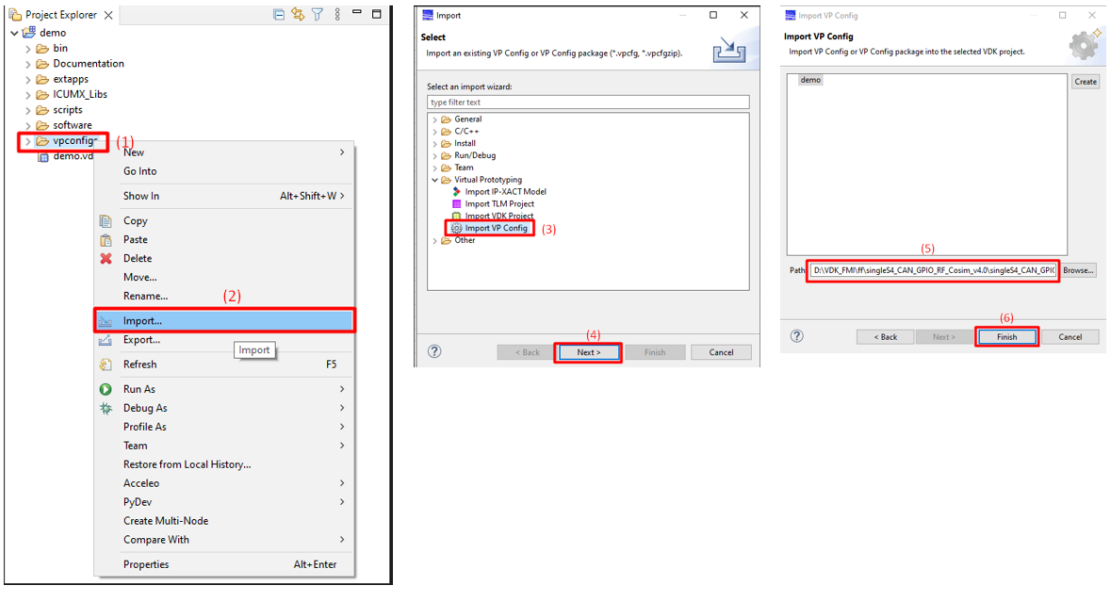


Figure 3-60 Show Project Explorer for R-Car V4H

➢ **Step 5:** Run simulation: Click [Run] button to start simulation.

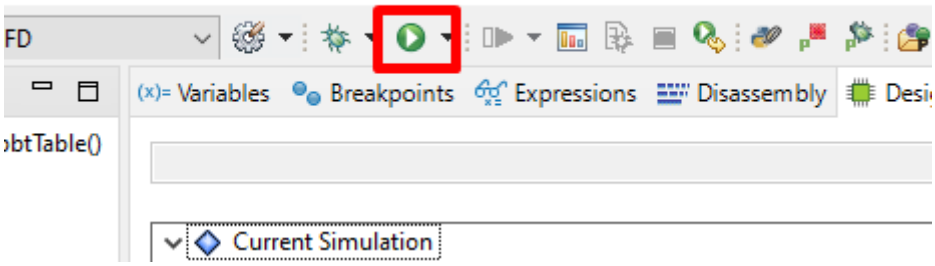


Figure 3-61 Importing available VP config file for R-Car V4H

3.4.5.2 Result of executing vHILS

(10) The vHILS execution starts on both the Simulink model and VDK.
Transmission and reception protocol between MATLAB and VDK is periodically after sampling time.
Remarks The TMU will request an interrupt each step time to invoke the <Code generation target>_step function toexchange data between VDK and MATLAB. The ETVPF will support config for TMU (for CR52 core) automatically and users do not config for it (Not Disclose).

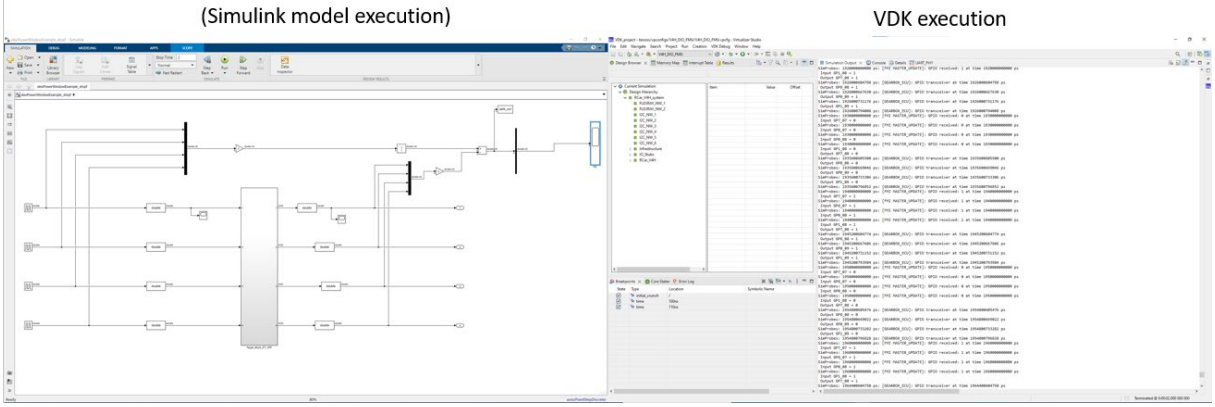


Figure 3-62 The vHILS execution *for* R-Car V4H

(11) The result of scope is below, MIL and ET-VPF results are the same.

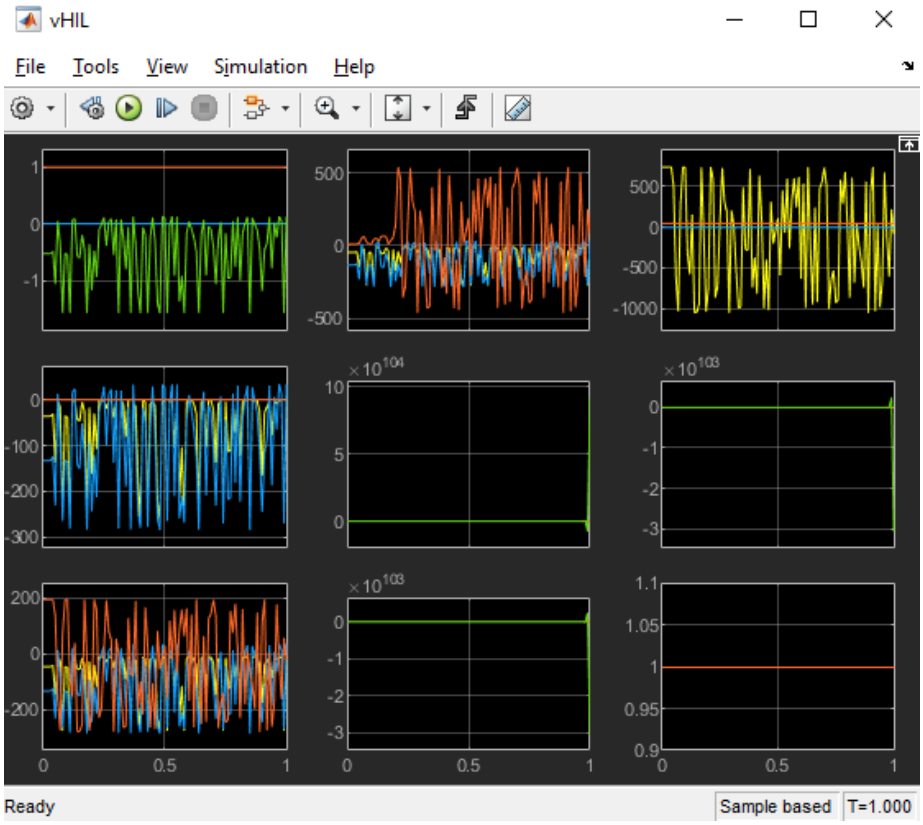


Figure 3-63 The result of scope for R-Car V4H

3.5 Time measurement

This section describes time measurement method which used in ET-VPF.

3.5.1 Structure of Simulink Model for measurement

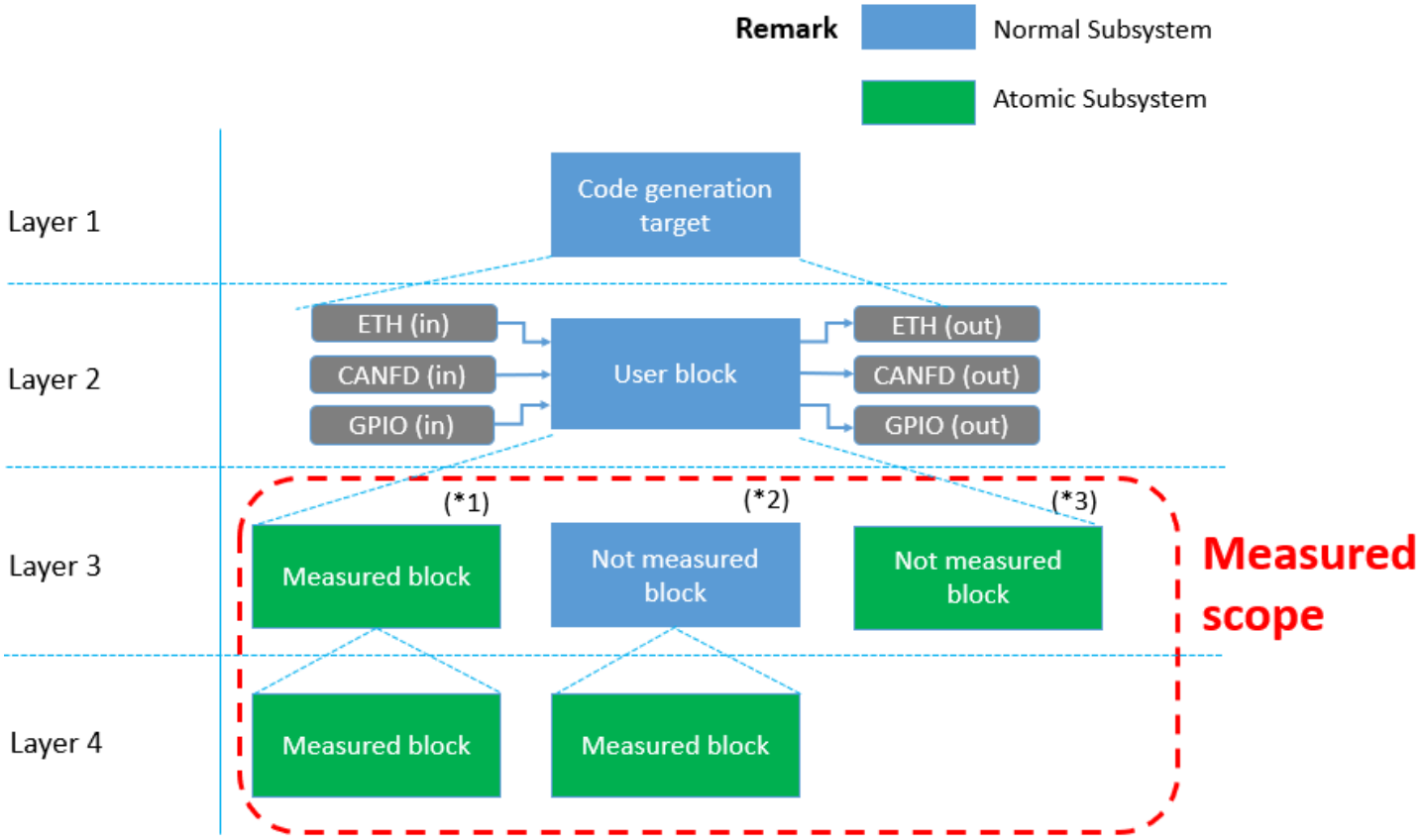


Figure 3-64 Structure of Simulink model for R-Car S4 measurement

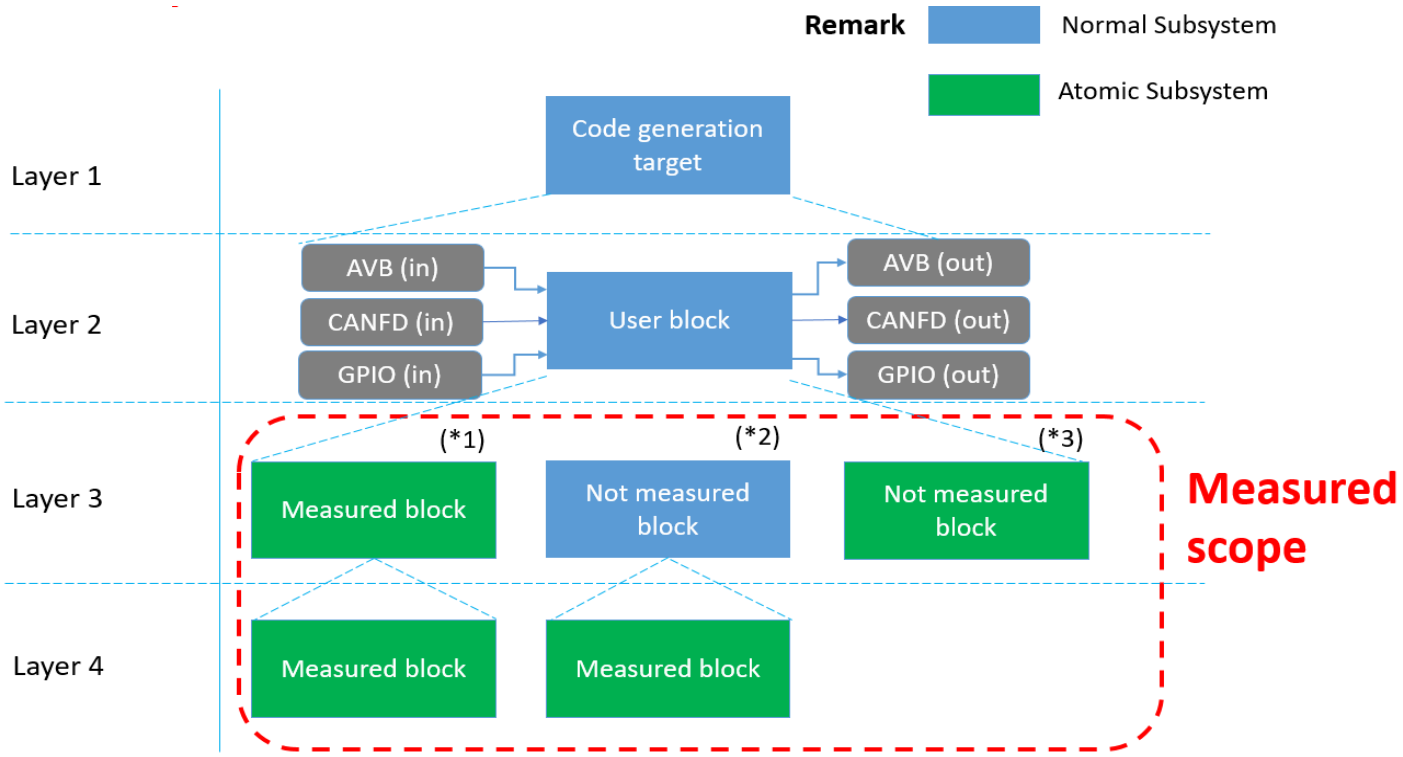


Figure 3-65 Structure of Simulink model for R-Car V4H measurement

Remarks (*1): The measured block meets all requirements for measurement.
(*2): User decision: Not measure this block. So, they don't need to transform it to Atomic Subsystem. Refer to section **3.6.3 How to change Normal Subsystem to Atomic Subsystem**.
(*3): Although it is Atomic Subsystem, but user does not define it in the input_subsystem.txt file.

Below is the requirement for the measured block:

1. Only measure for the block inside the User Block.
2. Only measure for the block that define in the input_subsystem.txt. Refer to section **3.6.2 Input file for Measurement**.
3. It must be the Atomic Subsystem.

3.5.2 Input file for Measurement

In the model folder, to decide which measured block, preparing the input_subsystem.txt with below format:

```
<Path of Atomic_Subsystem_name_1>,<First core>
<Path of Atomic_Subsystem_name_2>,<First core>
…
<Path of Atomic_Subsystem_name_N>,<First core>
```

For example:

Figure 3-66 Examples of input_subsystem.txt for Time Measurement for R-Car S4

3.5.3 How to change Normal Subsystem to Atomic Subsystem

To change the Normal Subsystem to Atomic Subsystem, do the following step:

(1) Right click on the Normal Subsystem -> Select Property
(2) In the Main tab, select the [treat as atomic unit] setting.
(3) In the Code Generation tab, set [Function packaging] to “Reusable function”.

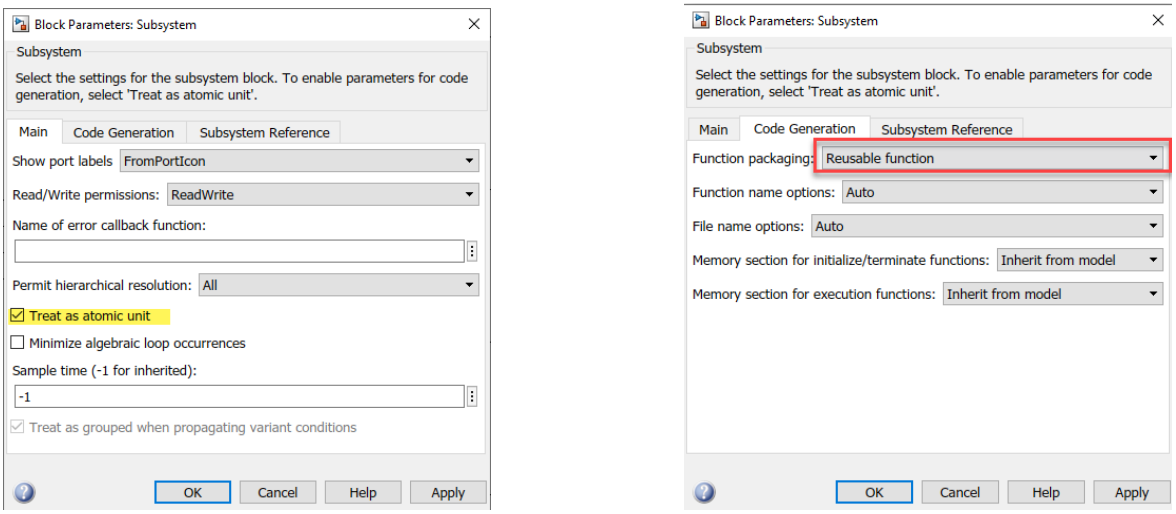


Figure 3-67 Change Normal Subsystem to Atomic Subsystem

3.5.4 Graph Viewer

The following describes how to display the time measurement results using Graph Viewer.

3.5.4.1 Input data

Graph Viewer has two input files:

Input 1: execution_data.csv file refers to upper data structure.

- Its unit is picosecond.
- It contains the execution time of each step and start end of each subsystem.

Target_Block_ET_VPF_step				
			Target_Block_ET_VPF_Subsystem	
Step	Valid	Time	Start	End
1	1	2469822	861894	861894
2	1	2490528	861894	861894
3	1	2490528	861894	861894
4	1	2490528	861894	861894
5	1	2490528	861894	861894
6	1	2491242	861894	861894
7	1	2491242	861894	861894
8	1	2499810	866178	866178
9	1	2491242	861894	861894
10	1	2491242	861894	861894
11	1	2490528	861894	861894
12	1	2490528	861894	861894
13	1	2490528	861894	861894
14	1	2490528	861894	861894
15	1	2490528	861894	861894
16	1	2499810	866178	866178
17	1	2491242	861894	861894
18	1	2491242	861894	861894
19	1	2491242	861894	861894
20	1	2491242	861894	861894

Input 2: input_subsystem.txt file that specifies the subsystem to be measured, refer to section **3.6.2 Input file for Measurement**.

3.5.4.2 Output figure

The following is the output figure that will be displayed when executing command of Graph Viewer (ETVPF_Launch_GraphViewer.m) in the MATLAB Command Window, using the following syntax.

Here ">>" denotes the command prompt and "[Enter]" denotes entry of the Enter key.

```
>> ETVPF_Launch_GraphViewer [Enter]
```

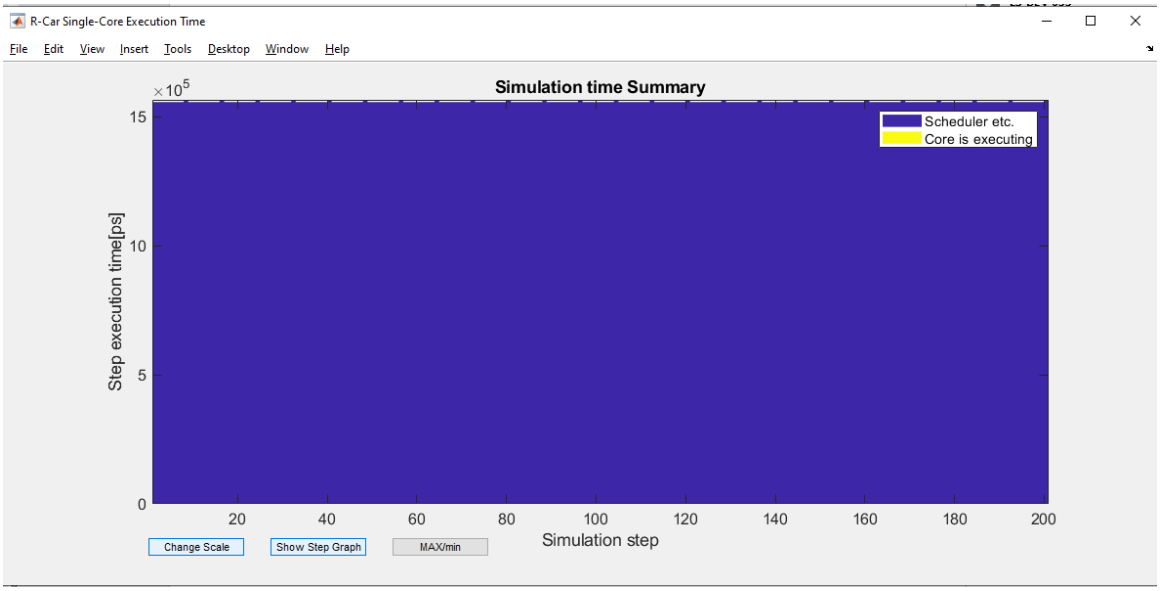


Figure 3-68 Output figure of time measurement

Remark Currently, the Graph Viewer do not have the execution time of peripheral source code. Users can perform some other functions as followings:

(1) Press [Show Step Graph] to shows execution time of each Subsystem for one step or more steps belonging to user choose in “Start Step” and “End Steps”.

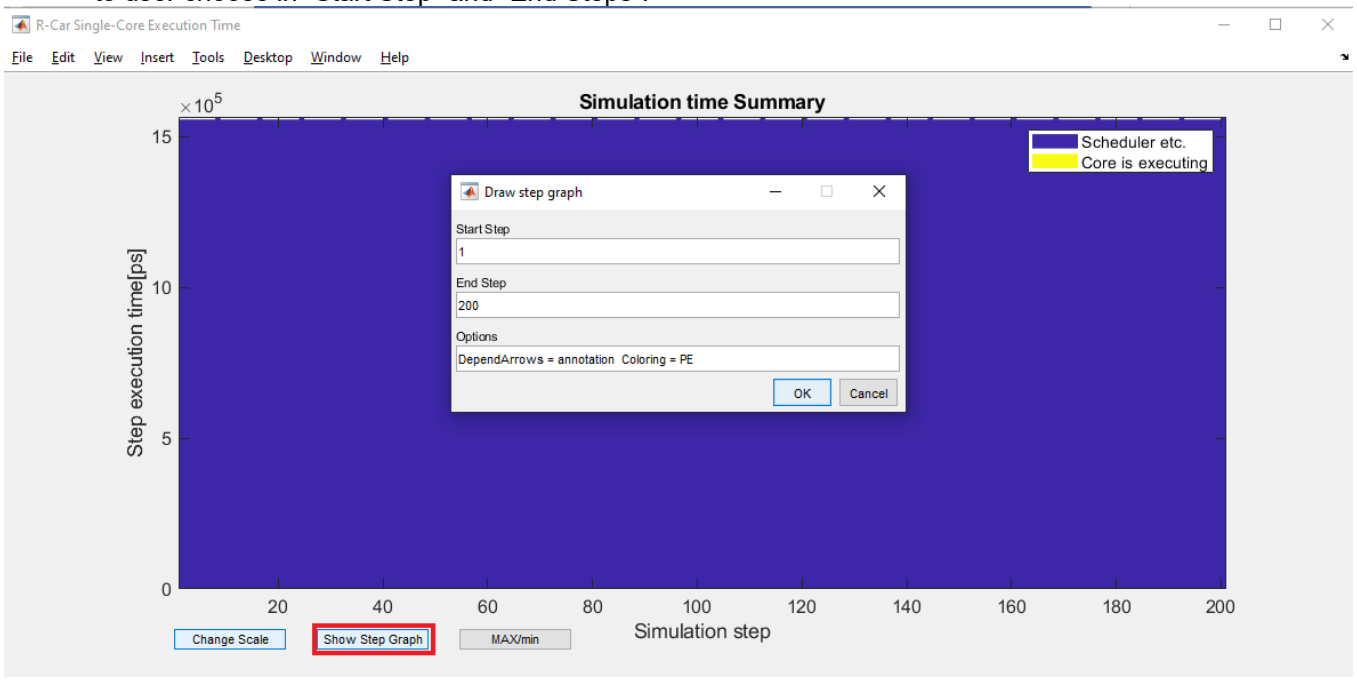


Figure 3-74 Press [Show Step Graph] to show execution time each Subsystem

Set “Start Step” and “End Start” (e.g., Start Step = 1, End Step = 1).

Figure 3-75 Example of Execution Time Detail at step 1

(2) Change scale of step by pressing [Change Scale] button then set “Start Step” and “End Start” (e.g., Start Step = 1, End Step = 100).

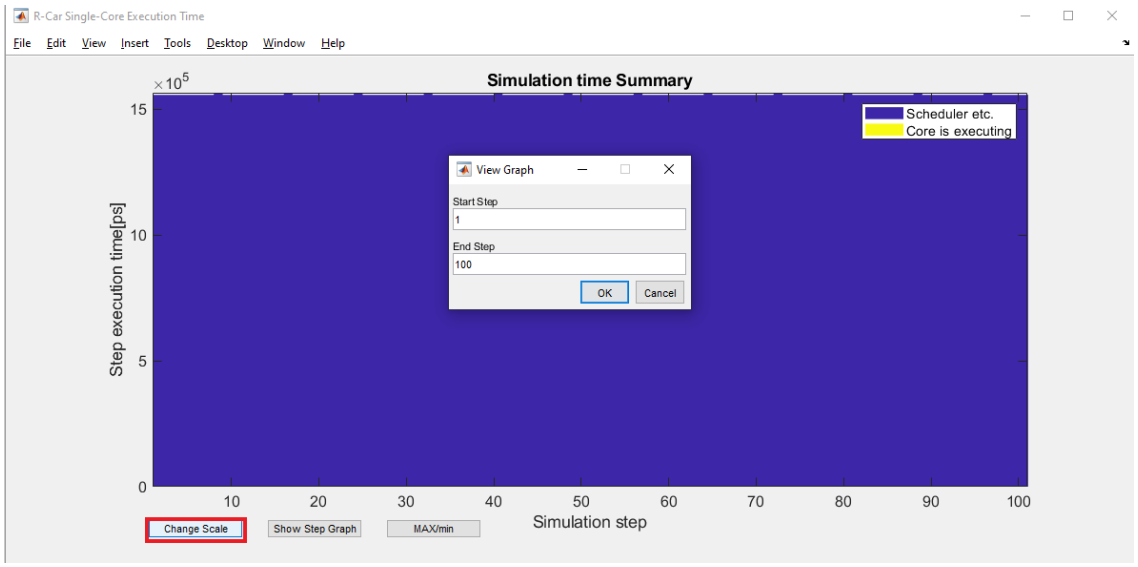


Figure 3-76 Press [Change Scale] to change scale of step

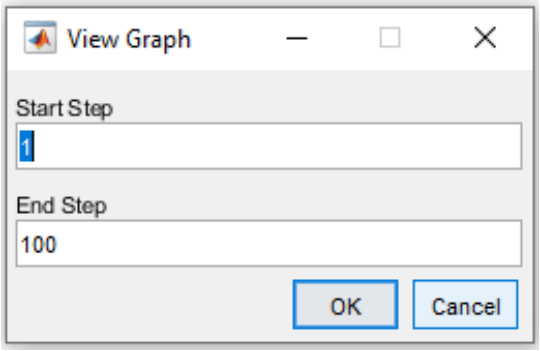


Figure 3-77 Set “Start Step” and “End Step” value

Then pressing [OK], the graph scale will change accordingly.

(3) Press [MAX/min] to shows which steps has the maximum execution time and minimum execution time.

Figure 3-78 Show Max/Min execution time

Then the Max Min Dialog show step has execution time maximum/minimum and the value of execution time of this step.
Press [Show step graph] in Maximum/Minimum Section to shows execution time detail of this step.

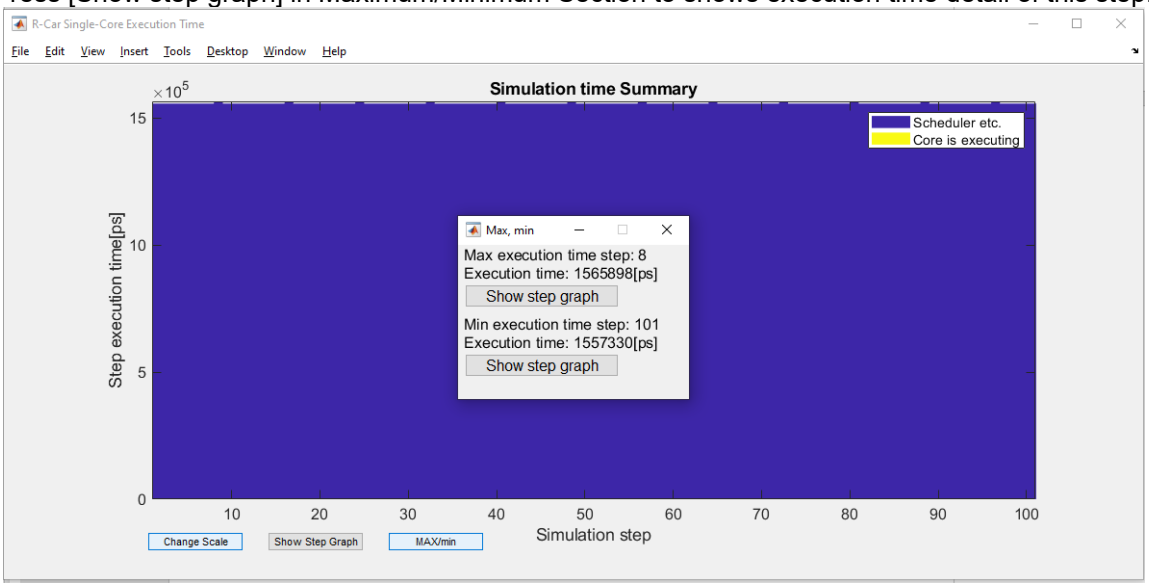


Figure 3-79 Max/Min Dialog

4 Points for Caution

This section describes limitations and points to note when using ET-VPF.

4.1 Features

The descriptions of affected features are removed in ET-VPF User's Manual.

(1) The name for measured blocks must be unique.
The current method to get the generated function for measured block is using the information in the comment. Its comment only has the Subsystem name. Then, cannot identify the target measured block by the Subsystem full path.

(2) To synchronize the communication between VDK (for R-Car S4 and R-Car V4H) and MATLAB, sampling time must higher than the execution time of algorithm each step. When the execution time is higher than sampling time, the output is not guaranteed. For specific execution time, it describes in the "Time" column of execution_data.csv in "<Code generation target>_etvpf" folder.

(3) The installation path of ET-VPF package, GHS (for R-Car S4), ARM (for R-Car S4 and R-Car V4H), Package MCAL source (for R-Car S4 and R-Car V4H), VDK (for R-Car S4 and R-Car V4H) cannot contain some special characters, refer to **Table 3-8 The supported special characters for R-Car S4**, **Table 3-13 The supported special characters for R-Car V4H**.

(4) For R-Car S4 and R-Car V4H, the communication between VDK and MATLAB, the output signal of ETVPF will be delayed by a step time compared with the MIL (refer to Delay a step time while co-simulating a FMU in simulink).

4.2 Simulink Models

4.2.1 Available Strings for Paths and Block Names

Do not use 2-byte characters (Japanese, etc.), spaces, slashes, line feeds, or hyphens for the names of code generation target blocks or paths to folders where the Simulink models are saved. If 2-byte characters are used for code generation target block names, they will be replaced with strings given by MATLAB, and if hyphens or spaces are used, their subsequent strings will be omitted. A series of ET-VPF operations is possible but not guaranteed.

4.2.2 Models Handling Complex Number Data

Code generation from the MATLAB/Simulink models (subsystems) handling complex number data is not supported.

4.2.3 Constant input of peripheral S-function block

When code generation from the MATLAB/Simulink models (subsystems) contains the constant output, MATLAB will generate unexpected source code. S-function source code for peripheral will be optimized.

To avoid this issue, please multiply the output of constant block with the “Repeating Sequence Stair” block (vector of output values is 1) in the user algorithm layer. After that connect the output to peripheral S-Function block. By this way, the output of constant block will be available every step with the same value.

4.3 Construction and Simulation

4.3.1 Length of Path to Code Generation Folder

The path length to files and directories is restricted to the Windows platform (limit of 260 characters). If the path length to the code generation folder containing source code generated by the Simulink model is too long, MATLAB will display an error message on MATLAB Command Window.

Error: Build failed because the build file name(s) exceeds the Windows limit of 260 characters. Build from a working directory with a shorter path, to allow build files to be created with shorter filenames. “<invalid file name>”

4.3.2 Notes on Power Management

If the PC goes to sleep or hibernate mode while working on ET-VPF, an error may occur during the operation after rebooting.

4.3.3 Length of Script File Name

For the length of script file name, it is limited by MATLAB. If its name is 64 characters or more, MATLAB will not allow executing it and an error will occur.

4.3.4 Install Drive and Work Drive

Make sure the ET-VPF installation drive and the working drive that stores the model are the same. Otherwise, an error will occur.

5 Error Messages

This section explains the error messages output by ET-VPF.

5.1 Overview

Error messages are output to notify you of information that you should know about events that occur while you are setting [ET-VPF options] in the Configuration Parameters dialog box or while a vHILS Simulation is running.

Remark Error messages output by ET-VPF are not linked to VDK. Therefore, no help is displayed even if you press the F1 key after ET-VPF displays an error message.

5.2 Errors Detected in Configuration Parameters Dialog Boxes

The following table lists the messages that are output when an error is detected while settings are being made in the Configuration Parameters dialog box.

These error messages are output to the ET-VPF Error dialog box.

Table 5-1 Error messages display when setting in the Configuration Parameters dialog box

[Message]	E0103 The VDK directory setting is incorrect.
[Explanation]	This error message is displayed when information on the correct installation destination has not been set in [VDK Installation Directory].
[Action by User]	1. Set the correct path of the installation destination of VDK to [VDK Installation Directory]. The folder containing “SLS/windows/setup.bat” must be specified. 2. Check that VDK has normally been installed.
[Message]	E0104 The GHS Compiler directory setting is incorrect.
[Explanation]	This error message is displayed when information on the correct installation destination has not been set in [GHS Compiler Installation Directory].
[Action by User]	1. Set the correct path of the installation destination of GHS to [GHS Compiler Installation Directory]. + For R-Car S4, the folder containing “ccrh850.exe” must be specified. 2. Check that GHS has normally been installed.

Table 5-2 Error messages display when setting in the Configuration Parameters dialog box (cont)

[Message]	E0105 The default directory is selected as the current installation directory. Deselect the use of the default directory.
[Explanation]	1. This error message displayed when [Select GHS Compiler Installation Directory] button is clicked, while the [Use default GHS Compiler Installation Directory] checkbox is checked. 2. This error message is displayed when the [VDK Installation Directory] button is clicked, while the [Use default VDK Installation Directory] checkbox is checked. 3. This error message is displayed when [MCAL Package Installation Directory] button is clicked, while the [Use default MCAL Package Installation Directory] checkbox is checked. 4. This error message is displayed when [ARM Compiler Installation Directory] button is clicked, while the [Use default ARM Compiler Installation Directory] checkbox is checked
[Action by User]	1. Uncheck [Use default GHS Compiler Installation Directory] checkbox. 2. Uncheck [Use default VDK Installation Directory] checkbox. 3. Uncheck [Use default MCAL Package Installation Directory] checkbox. 4. Uncheck [Use default ARM Compiler Installation Directory] checkbox.
[Message]	E0106 The ARM Compiler directory setting is incorrect.
[Explanation]	This error message is displayed when information on the correct installation destination has not been set in [ARM Compiler Installation Directory].
[Action by User]	1. Set the correct path of the installation destination of ARM to [ARM Compiler Installation Directory]. The folder containing “armclang.exe” and “armlink” must be specified. 2. Check that ARM has normally been installed.
[Message]	E0107 The MCAL Package Installation directory setting is incorrect.
[Explanation]	This error message is displayed when information on the correct installation destination has not been set in [MCAL Package Installation Directory].
[Action by User]	1. Set the correct path of the installation destination of MCAL Package to [MCAL Package Installation Directory]. + For R-Car S4, the folder containing “g4mh_mcal" and "cr52_mcal" folders must be specified. + For R-Car V4H, the folder containing “rel/V4H/common_family/make/arm/SampleApp.bat” must be specified. 2. Check that MCAL Package has normally been installed.

Table 5-3 Error messages display when setting in the Configuration Parameters dialog box (cont)

[Message]	E0110 A license is not registered.
[Explanation]	This message displays when no license or license was expired on your system.
[Action by User]	Register ET-VPF License with Renesas Electronics Corporation.
[Message]	E0111 <Device Series> is not available. Register a valid license.
[Explanation]	This error message displayed when either the license for any <Device Series> is invalid.
[Action by User]	1. If you don't have the license for <Device Series>, please contact Renesas Electronics sales office. 2. If you have got license <Device Series>, check if it is put in the ET-VPF installation. 3. To confirm the availability of the license, please [Check Available License] on [ET-VPF options] panel.

5.3 Errors during vHILS execution

The following describes error messages detected when you get visual display of the execution time in algorithm verification after executed vHILS. Error dialog boxes during vHILS are output from MATLAB/Simulink

Table 5-4 Error messages display when getting the execution time

[Message]	E0201 Wrong measurement data. Please check data input files.
[Explanation]	The execution time measurement result is wrong or empty.
[Action by User]	1. Check if the model settings and components are correctly. 2. If above information is correct, re-execute vHILS to generate execution time measurement result again.

5.4 Errors during building and generating the vHILS environment

The following table lists the messages that are detected in ET-VPF processing when building and generating the vHILS environment.

Table 5-5 Error messages display when building and generating the vHILS environment

[Message]	E0310 The <Full File Name> file does not exist. Please check this file.
[Explanation]	This error message is displayed when cannot find the input file.
[Action by User]	1. Check if the input file has existed or not. 2. Check if the full path of input file is right or not.
[Message]	E0302 The <Full File Name> file is conflict.
[Explanation]	This error message displays when different configured functions are simultaneously in one port.
[Action by User]	1. Check the pin configuration in the User's Manual Hardware 2. Check that the GPIO corresponding to <Peripheral Device 1> being used overlaps with the <Peripheral Device 2> configured in the model.
[Message]	E0303 The <Core Name> don't support ethernet <ETH Unit Name> unit(s).
[Explanation]	This error message displays when ETHERNET functions are configured incorrectly for each core.
[Action by User]	Configure Ethernet functions corresponding to each function that the core supports. 1. AVB-IF function is supported by core G4MH. 2. RSW2 function is supported by core CR52.

5.5 Errors during communication between MATLAB and VDK

The following table lists the messages that are detected in ET-VPF processing when communication between MATLAB and VDK.

Table 5-6 Error messages display when building and generating the vHILS environment

[Message]	E0401 An incorrect automation server is registered.
[Explanation]	A different version of MATLAB registered as the Automation Server.
[Action by User]	Use “regmatlabserver” command (without quotes) to register current MATLAB as an Automation Server.

Index

REVISION HISTORY		Embedded Target for R-Car Virtual Platform V1.00.00 R-Car Model-Based Development Tool User's Manual	
Rev.	Date	Description	
		Page	Summary
1.00	January 16, 2024	-	First edition issued

Embedded Target for R-Car Virtual Platform
V1.00.00

R-Car Model-Based Development Tool User's Manual

Publication Date: Rev.1.00 January 16, 2024

Published by: Renesas Electronics Corporation

Embedded Target for R-Car Virtual Platform V1.00.00

R-Car Model-Based Development Tool

Renesas Electronics Corporation

R20UT5334EJ0100