|  |  |
| --- | --- |
| T(Title)  **System Specifications for**  **ET-VPF®**  **(Embedded Target for Virtual Platform)**  **V1.00.00** | (Created by)  Core Software 3 Section,  Core Software Department,  Software Engineering Division,  Renesas Design Vietnam Co., Ltd.  (Modified by)  Software Engineering Division,  Renesas Design Vietnam Co., Ltd. |

This document describes the system specifications of Embedded Target for Virtual Platform (hereafter referrerd to as ET-VPF).

~~# Lines with Strike through are not used anymore.~~

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Revision Number** | **Description** | **Approved** | **Checked** | **Created** |
| 1 | R01 | Update the contents for RLIN3n  Update the contents of F1KM-S4 |  | Phuc Giang  Hong Tieu  Giang Nguyen  Son Thai  Tinh Le  Jun 28, 2022 | Hong Tieu  Giang Nguyen  Jun 27, 2022 |
| 2 | R02 | Update the contents for U2C alpha version |  | Phuc Giang  Hong Tieu  Giang Nguyen  Son Thai  Tinh Le  Jul 07, 2022 | Hong Tieu  Son Thai  Jul 01, 2022 |
| 3 | R03 | Update the contents for TAUD  Update the contents for RS-CANFD |  | Phuc Giang  Hong Tieu  Giang Nguyen  Son Thai  Tinh Le  Jul 25, 2022 | Tinh Le  Jul 11, 2022 |
| 4 | R04 | Update the contents after fixed bug |  | Phuc Giang  Hong Tieu  Giang Nguyen  Tinh Le  Sep 15, 2022 | Tinh Le  Phuc Giang  Sep 12, 2022 |
| 5 | R05 | Update the contents after fixed REL’s comments |  | Phuc Giang  Hong Tieu  Son Thai  Tinh Le  Sep 19, 2022 | Phuc Giang  Sep 19, 2022 |
| 6 | R06 | Update the contents of U2C alpha version after received new toolbox |  | Phuc Giang  Hong Tieu  Tinh Le  Giang Nguyen  Sep 23, 2022 | Son Thai  Sep 22, 2022 |
| 7 | R07 | Update the contents for ET-VPF installer |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen  Oct 25, 2022 | Hong Tieu  Oct 17, 2022 |
| 8 | R08 | Update the installation procedure |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen  Nov 01, 2022 | Hong Tieu  Oct 31, 2022 |
| 9 | R9 | Update ADC unit 1 for RH850/F1KM-S4 |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen  Nov 09, 2022 | Giang Nguyen  Nov 02, 2022 |
| 10 | R10 | Update peripheral S-function blocks to MATLAB Simulink Library Browser |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen  Nov 28, 2022 | Giang Nguyen  Nov 25, 2022 |
| 11 | R11 | Update the file list of Renesas Electronics MCU Tools |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen  Dec 06, 2022 | Hong Tieu  Dec 06, 2022 |
| 12 | R12 | Update MATLAB R2021a contents |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen  Dec 12, 2022 | Tinh Le  Dec 02, 2022 |
| 13 | R13 | Add CAN for U2C Device Series |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen  Dec 22, 2022 | Phuc Giang  Nov 29, 2022 |
| 14 | R14 | Update for Time Measurement Method |  | Phuc Giang  Hong Tieu  Tinh Le  Son Thai  Giang Nguyen | Son Thai  Dec 19, 2022 |

RELATED DOCUMENTS

|  |  |
| --- | --- |
| **Document Name** | **Document No.** |
| CS+全体システム仕様書  (CS + Overall System Specification) | RSO-14-004703 |
| CS+用DLLインターフェース仕様書 デバッガ-GUI拡張部  (CS + DLL interface specification Debugger - GUI extension) | RSO-14-003974 |
| デバッガ システム仕様書 GUI部  (Debugger system specification GUI section) | RSO-14-004340 |
| CS+ Python関数仕様書  (CS + Python function specification) | RSO-14-004389 |
| ルネサス エレクトロニクス マイコン開発ツール ライセンス インタフェース仕様書  (Renesas Electronics Microcomputer Development Tool License Interface Specification) | LLWEB-10007091 |
| Simulinkモデル変形スクリプト生成ツール開発 (機能仕様書)  (Simulink Model Conversion Script Generation Tool Development Function Specification) | - |
| RH850\_Toolbox\_F1x\_Addendum.pdf | - |
| RH850\_Virtual\_Platform\_Module\_Reference.pdf | - |
| VLAB\_RH850\_Virtual\_Platform\_Toolbox\_User\_Manual.pdf | - |
| Smart Configurator User’s Manual RH850 API Reference | - |

TABLE OF CONTENTS

[1 GENERAL 6](#_Toc122608900)

[1.1 Overview 6](#_Toc122608901)

[1.2 Operating Environment 7](#_Toc122608902)

[1.3 Target Device Series 8](#_Toc122608903)

[1.4 License Policy 8](#_Toc122608904)

[1.5 Package to be installed 9](#_Toc122608905)

[2 INSTALLATION 11](#_Toc122608906)

[2.1 Installation 11](#_Toc122608907)

[2.2 Uninstallation 13](#_Toc122608908)

[3 FUNCTIONS 15](#_Toc122608909)

[3.1 Overview 15](#_Toc122608910)

[3.2 S-Function block of peripherals 16](#_Toc122608911)

[3.2.1 ADC peripheral 19](#_Toc122608912)

[3.2.2 Port peripheral 21](#_Toc122608913)

[3.2.3 RS-CANFD peripheral 23](#_Toc122608914)

[3.2.4. RLIN3n peripheral 25](#_Toc122608915)

[3.2.5 TAUD peripheral 26](#_Toc122608916)

[3.3 Executing Simulator Processor in the Loop Simulation 29](#_Toc122608917)

[3.3.1 Embedded sample model 29](#_Toc122608918)

[3.3.2 Setting configuration parameters 31](#_Toc122608919)

[3.3.3 Generating a SPILS environment 37](#_Toc122608920)

[3.3.4 Compiling generated source code 42](#_Toc122608921)

[3.3.5 Executing SPILS 43](#_Toc122608922)

[3.4 Time measurement 45](#_Toc122608923)

[3.4.1. Structure of Simulink Model for measurement 45](#_Toc122608924)

[3.4.2. Input file for Measurement 46](#_Toc122608925)

[3.4.3. How to change Normal Subsystem to Atomic Subsystem 46](#_Toc122608926)

[3.4.4 Graph Viewer 47](#_Toc122608927)

[4 POINTS FOR CAUTION 52](#_Toc122608928)

[4.1 Features 52](#_Toc122608929)

[4.2 Simulink models 53](#_Toc122608930)

[4.2.1 Available Strings for Paths and Block Name 53](#_Toc122608931)

[4.2.2 Models Handling Complex Number Data 53](#_Toc122608932)

[4.3 Construction and Simulation 53](#_Toc122608933)

[4.3.1. Length of Path to Code Generation Folder 53](#_Toc122608934)

[4.3.2 Notes on Power Management 53](#_Toc122608935)

[4.3.3 Length of Script File Name 53](#_Toc122608936)

[4.3.4 Install Drive and Work Drive 53](#_Toc122608937)

[5 ERROR MESSAGES 54](#_Toc122608938)

[5.1 Overview 54](#_Toc122608939)

[5.2 Errors Detected in Configuration Parameters Dialog Box 54](#_Toc122608940)

[5.3 Errors during SPILS execution 56](#_Toc122608941)

[6 CHANGE HISTORY 57](#_Toc122608942)

# GENERAL

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

## 1.1 Overview

Embedded Target is used to check the function and measurement issue by running the code, which is considered on MILS on the device. (MILS-PILS back-to-back verification).

But at the phase to consider the algorithm, there is no environment to measure the performance including the peripherals.

In the meantime, if the time is estimated with peripherals, need to connect driver code to the code generated from the application and it is a high effort for OEM who doesn’t know the MCU.

ET-VPF is a tool that support users simulating the MATLAB Simulink block (with peripherals) on a virtual Hardware (Renesas RH850 virtual platform). ET-VPF uses Co-simulation toolbox to communicate between MATLAB Simulink and ASTC VLAB.

The purpose is to estimate the time at early phase by preparing the code including the peripherals driver code verifying the function and estimating the performance by realizing the back-to-back verification b/w MILS – ET-VPF.

By supporting ET-VPF, Renesas can provide the new MBD solution to the customer rather than the other vendors. As this technology is applied to ET-VPF, Renesas should be able to support the HILS in the future. As of now, as there are many HILS users, this approach must be accepted by customers.

Graphical user interface, diagram, application

Description automatically generated

Diagram

Description automatically generated

**Figure 1‑1 Overview of Embedded Target for Virtual Platform‑**

## 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 V9.10 (R2021a), V9.3 (R2017b)

Simulink V10.3 (R2021a), V9.3 (R2017b)

Stateflow V10.4 (R2021a), V9.0 (R2017b)

MATLAB Coder V5.2 (R2021a), V9.0 (R2017b)

Simulink Coder V9.5 (R2021a), V8.13 (R2017b)

Embedded Coder V7.6 (R2021a), V6.13 (R2017b)

Vehicle Network Toolbox V5.0 (R2021a), V3.4 (R2017b)

* MEX-file compiler

Microsoft Visual C++ 2013, 2015, 2017 compiler (from Microsoft Corporation)

Reference: System Requirements & Platform Availability

<https://www.mathworks.com/support/sysreq/previous_releases.html>

* VLAB (from The VLAB Works Pty Ltd)

VLAB V2.7.2 (win-vc140-x64)

* Toolboxes of VLAB (included with VLAB V2.6.1)

Toolbox RH850 Virtual Platform V3.1.8 (For test supported devices (RH850/F1M-S1, RH850/F1M-S4))

Toolbox CAN 2.5.0 (For test supported peripheral RS-CANFD)

Toolbox RH850 G4 Virtual Platform 1.16.0 (For test supported devices RH850/U2C)

* Smart Configurator for RH850 (from Renesas Electronics Corp)

SC V1.5.0

* Build Tool

CC-RH Included with CS+ V8.07.00/ E8.07.00j2 (from Renesas Electronics)

* Cygwin (from Cygwin Authors)

Cygwin V2.11.2

* Nullsoft Scriptable Install System Tool

NSIS V3.08

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 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. MATLAB R2017b is used because when preparing the prototype of vHILS, the VLAB Co-Simulink supports R2017b. For MATLAB R2021a, the VLAB Co-Simulink does not confirm the supports. For other versions, they have not been tested.

4. Can use the upper software versions if the features used for ET-VPF be not changed but Renesas Electronics does not guarantee it.

5. For CS+ E8.07.00j2, it is only used for RH850/U2C device series. For CS+ V8.07.00, it is only used for RH850/F1KM-S1, RH850/F1KM-S4 device series.

6. The Smart Configurator for RH850 is unavailable for RH850/U2C device series.

7. The installation path of ET-VPF package, CS+, Cygwin, Smart Configurator and VLAB cannot contain some special characters (refer to **Table 3‑6 The supported special characters** for more detail).

## 1.3 Target Device Series

The following devices are supported in ET-VPF.

**Table 1‑1 Supported devices**

|  |  |
| --- | --- |
| **Series** | **Devices** |
| RH850/F1x | RH850/F1KM-S1, RH850/F1KM-S4 |
| RH850/U2x | RH850/U2C |

## 1.4 License Policy

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

The table below shows available operation when we own each license type.

**Table 1‑2 License definition of ET-VPF**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **License name** | **Supported device** | | | **Time measurement feature** | **Supported build tool** | |
| **RH850/F1KM-S1** | **RH850/F1KM-S4** | **RH850/U2C** | **GHS Compiler** | **Renesas Compiler** |
| Embedded Target for Virtual Platform RH850 | √ | √ | √ | √ | - | √ |

\* √ : Can execute  
 -: Cannot execute

## 1.5 Package to be installed

The following table describes the structure of ET-VPF package for RH850/F1x device series after installing successfully.

**Table 1‑3 ET-VPF release package definition for RH850/F1x device series**

|  |  |  |  |
| --- | --- | --- | --- |
| **Installed package** | | | **Description** |
| <ET-VPF installation folder>\<version information>\F1x\ETVPF\_package | ETVPF\_include | ETVPF\_S\_function\_block | Contains the source code which used to execute the program of peripherals. |
| make | Contains the template of make file. |
| Renesas   * Common\_files\_F1KM\_S1 * Common\_files\_F1KM\_S4 * make | Contains the source code of each device which used for Renesas compiler.  ***Note:*** User can modify boot code files in “startup” folder, but Renesas Electronics does not take responsibility for quality. |
| XML\_input   * RH850   + F1KM-S1   + F1KM-S4 | Contains the XML input files which provided for each RH850 device series. |
| Source code files | Contains the list of source code files (include: \*.p, \*.tlc, make files) which used to execute the program of ET-VPF. |
| <ET-VPF installation folder> | Uninst\_ETVPF\_<version information>.exe | | Uninstallation file used to uninstall ET-VPF package |
| C:\Program Files (x86)\Common Files\Renesas Electronics MCU Tools | ToolInfo | - | Contain the following file:   * Mjywm4.dat |
| VHILS | Contain the following file:   * \_vh850c1x\_na * \_vh850e1x\_na * \_vh850e2x\_na * \_vh850f1x\_na * \_vh850p1x\_na * \_vh850u2x\_na |

The following table describes the structure of ET-VPF package for RH850/U2x device series after installing successfully.

**Table 1‑4 ET-VPF release package definition for RH850/U2x device series**

|  |  |  |  |
| --- | --- | --- | --- |
| **Installed package** | | | **Description** |
| <ET-VPF installation folder>\<version information>\U2x\ETVPF\_package | ETVPF\_include | ETVPF\_S\_function\_block | Contains the source code which used to execute the program of peripherals. |
| make | Contains the template of make file. |
| Renesas   * Common\_files\_U2C * make | Contains the source code of each device which used for Renesas compiler.  ***Note:*** User can modify boot code files in “startup” folder, but Renesas Electronics does not take responsibility for quality. |
| Source code files | Contains the list of source code files (include: \*.p, \*.tlc, make files) which used to execute the program of ET-VPF. |
| <ET-VPF installation folder> | Uninst\_ETVPF\_<version information>.exe | | Uninstallation file used to uninstall ET-VPF package |
| C:\Program Files (x86)\Common Files\Renesas Electronics MCU Tools | ToolInfo | - | Contain the following file:   * Mjywm4.dat |
| VHILS | Contain the following file:   * \_vh850c1x\_na * \_vh850e1x\_na * \_vh850e2x\_na * \_vh850f1x\_na * \_vh850p1x\_na * \_vh850u2x\_na |

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 located in the same <system drive> with the user’s workspace to avoid the error occurring when loading the loaded module to RH850 virtual platform.

3. The <ET-VPF installation folder> must not contain the special characters (refer to **Table 3‑6 The supported special characters**). An error will occur when executing ET-VPF if <ET-VPF installation folder> has the special characters.

# INSTALLATION

This section describes the installation and uninstallation procedure for the ET-VPF package.

## 2.1 Installation

To create Installer for ET-VPF, use Nullsoft Scriptable Install System (NSIS) Tool to compile NSI scripts to installation file (\*.exe file) as below steps:

* **Step 1:** Download and install Nullsoft Scriptable Install System (NSIS) Tool.
* **Step 2:** Create NSI scripts to define ET-VPF Installer.
* **Step 3:** Prepare the source packages (contain: ETVPF\_package, Renesas Electronics MCU Tools folders) for ET-VPF Installer.
* **Step 4:** Load NSI scripts by NSIS Tool to create ET-VPF Installer.

The following installation file is necessary to install ET-VPF.

* ETVPF\_<version information>\_Setup.exe

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.

After install ET-VPF, the programs, library and files list are in the folders structure which described in the section **1.5 Package to be installed**.

Remarks 1. Currently, ET-VPF V1.00.00 installer only support to install ET-VPF package of RH850/F1x device series.

2. The current Product Name is “Embedded Target for RH850 Virtual Platform”.

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

4. To avoid the warning when downloading the setup file from the website, a zip file for ET-VPF installer (ETVPF\_<version information>\_Setup.7z) will be prepared.

The following describes the procedure for installing ET-VPF.

1. Install the ET-VPF package by the below steps:

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

Graphical user interface, application, Teams

Description automatically generated

**Figure 2‑1 Installation Procedure**

* **Step 4:** Check installed package follow section **1.5 Package to be installed**. If the package installed by ET-VPF Installer is as same as the description in section **1.5 Package to be installed**, this proves that the installation was successful.

1. After installing the ET-VPF package, the user must set the 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

Graphical user interface, text, application

Description automatically generated

**Figure 2‑2 Adding the ET-VPF folder**

1. 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 we change the in-use MATLAB version, execute this command again.

1. Setup the MATLAB Simulink Library Browser

Execute the following command from the MATLAB command window to setup and add ETVPF package into the MATLAB Simulink Library Browser.

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

>> setup\_etvpf\_lib [Enter]

## 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 **2.1 Installation**) on the [Set Path] dialog box.

Graphical user interface, text, application

Description automatically generated

**Figure 2‑3 Removing the ET-VPF folder**

1. Uninstall the ET-VPF package as below steps:

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

Graphical user interface, text, application

Description automatically generated

**Figure 2‑4 Uninstallation Confirmation Message**

A screenshot of a computer

Description automatically generated

**Figure 2‑5 Uninstallation Procedure**

* **Step 3:** Check “<ET-VPF installation folder>\<version information>” folder and “Uninst\_ETVPF\_<version information>.exe” are removed. If they are deleted, uninstalling ET-VPF package is successful.

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

1. 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]

Remark If (3) is performed prior to (1) above, a warning will be displayed the next time MATLAB is started.

# FUNCTIONS

This section describes the functions provided by ET-VPF.

## 3.1 Overview

ET-VPF provides the functions to generate a SPILS environment and to verify algorithms (prepare environment within 10 minutes). ET-VPF generates a SPILS environment in cooperation with Embedded Coder.

The following describes the main procedure of ET-VPF:

* Set configuration parameters
* Clean existed files, objects.
* Generate source code files.
* Generate C source files of peripherals (Port, ADC, RLIN3n, TAUD) (using SC to generate them).
* Generate source files of target device.
* Generate C source files.
* Generate make file and config file.
* Generate application make file.
* Generate VLAB execution file.
* Generate python files.
* Generate Define.h, OSTM\_define.h.
* Call Cygwin to build source code.
* Start-up VLAB then run the target python script.
* Wait until the co-sim finish and verify the result.

About S-Function block of peripherals, there are some sample S-Function blocks (Port, ADC, RS-CANFD, RLIN3n, TAUD) included in “etvpf\_lib.slx” file, which is provided by Renesas Electronics, user can use these available S-Function blocks.

ET-VPF will support generate a SPILS environment and verify algorithms automatically. But the settings of peripherals are dependent on each use case. Therefore, ET-VPF also support user do it by manual: during generation of SPILS environment, user can configure settings, generate source code for peripherals (Port, ADC, RLIN3n, TAUD) easier, more detail via SC. To do this step, user can refer to section **3.3.3.1 Generating the peripherals’ source code by SC**.

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

***Table 3‑1 List of usable blocks***

|  |  |  |
| --- | --- | --- |
| **No** | **Usable blocks** | **Configurable blocks** |
| 1 | Subsystem | Graphical user interface, text, application  Description automatically generated with medium confidence |
| 2 | Inport (\*1) | Diagram  Description automatically generated |
| 3 | Outport (\*1) | Diagram  Description automatically generated |
| 4 | Mux | Icon  Description automatically generated |
| 5 | Demux | A black letter on a white background  Description automatically generated with low confidence |
| 6 | Data Type Conversion | Text  Description automatically generated with medium confidence |
| 7 | ADC | A picture containing text  Description automatically generated |
| 8 | PORT | A picture containing table  Description automatically generatedA picture containing shape  Description automatically generated |
| 9 | TAUD | Text, whiteboard  Description automatically generated |
| 10 | RLIN3 | Diagram  Description automatically generated with medium confidenceDiagram  Description automatically generated with medium confidence |
| 11 | RS-CANFD | Text  Description automatically generatedText  Description automatically generated with medium confidence |
| 12 | Chart | Text, icon  Description automatically generated |

(\*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 user to use the Inport shadow block.

## 3.2 S-Function block of peripherals

The following describes the S-Function blocks of peripherals (Port, ADC, RS-CANFD, RLIN3n, TAUD), and 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, user must select the correct peripherals’ settings with the current device series (peripherals’ settings on model must be the same with the settings on SC – refer to section **3.3.3.1 Generating the peripherals’ source code by SC** to know the configuration method). 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 RH850/F1KM, only use the following settings.

1. For ADC peripheral: support for Scan Group 1, Multicycle scan mode.
2. For Port peripheral: Only can use normal port (PORT) (e.g., PORT0, PORT1… defined in SC).
3. For RS-CANFD peripheral:

For F1KM: Support use CANFD unit 0 (RCFDC0) with 6 channel, Sending/receiving Data frame, Standard ID, the remote frame, extended ID, receiving filter, Support CAN Port I/O and Vehicle Network Toolbox.

For U2C: Support use CANFD unit 0 (RCFDC0) with 8 channel, Sending/receiving Data frame, Standard ID, the remote frame, extended ID, receiving filter, and Vehicle Network Toolbox.

1. For TAUD peripheral: Only can use PWM Output and Triangle PWM Output, master channel, pulse cycle and duty ratio value settings. About slave channels, and clock setting please set it in SC

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.

4. About the peripherals used for RH850/U2C, only use the Port, ADC and CAN peripherals.

The following table shows the information about the peripherals of each device series that is supported for S-Function blocks. For S-Function blocks of each device series, only select settings as described in this table.

**Table 3‑2 Peripherals settings for sample S-Functions of each device series (Not open to users)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Device series** | | **RH850/F1KM-S1** | **RH850/F1KM-S4** | **RH850/U2C** |
| ADC | Max channel | 36 | 36 | 28 |
| ADC port name | ANI000 -> ANI035 | ANI000 -> ANI027  ANI030 -> ANI035  ANI100 -> ANI135 | AN100 -> AN103  AN110 -> AN113  AN120 -> AN123  AN130 -> AN133  AN140 -> AN143  AN150 -> AN153  AN160 -> AN163  AN170 -> AN173  AN180 -> AN181 |
| Port | Port name | P00\_0 -> P00\_14  P08\_0 -> P08\_12  P09\_0 -> P09\_6  P10\_0 -> P10\_15  P11\_0 -> P11\_7 | P00\_0 -> P00\_14  P01\_0 -> P01\_5, P01\_8 -> P01\_15  P02\_0 -> P02\_15  P03\_0  P08\_0 -> P08\_12  P09\_0 -> P09\_4  P10\_0 -> P10\_15  P11\_0 -> P11\_12, P11\_15  P12\_0 -> P12\_5  P13\_0 -> P13\_7  P18\_0 -> P18\_15  P19\_0 -> P19\_3  P20\_0 -> P20\_5 | P02\_0 -> P02\_14  P03\_0, P03\_1, P03\_10, P03\_11, P03\_12  P04\_0 -> P04\_15  P06\_0 -> P06\_13  P10\_0 -> P10\_14  P17\_0 -> P17\_13  P20\_0 -> P20\_13  P21\_0 -> P21\_10  P22\_0 -> P22\_9  P24\_4 -> P24\_13 |
| RS-CANFD | Port name | CAN0 RX (P0\_1)  CAN0 TX (P0\_0)  CAN1 RX (P0\_2)  CAN1 TX (P0\_3)  CAN2 RX (P0\_5)  CAN2 TX (P0\_4)  CAN3 RX (P0\_7)  CAN3 TX (P0\_8)  CAN4 RX (P0\_9)  CAN4 TX (P0\_10)  CAN5 RX (P0\_13)  CAN5 TX (P0\_14) | CAN0 RX (P0\_1)  CAN0 TX (P0\_0)  CAN1 RX (P0\_2)  CAN1 TX (P0\_3)  CAN2 RX (P0\_5)  CAN2 TX (P0\_4)  CAN3 RX (P0\_7)  CAN3 TX (P0\_8)  CAN4 RX (P0\_9)  CAN4 TX (P0\_10)  CAN5 RX (P0\_13)  CAN5 TX (P0\_14) | CAN0 RX (T.B.D)  CAN0 TX (T.B.D)  CAN1 RX (T.B.D)  CAN1 TX (T.B.D)  CAN2 RX T.B.D)  CAN2 TX (T.B.D)  CAN3 RX (T.B.D)  CAN3 TX (T.B.D)  CAN4 RX (T.B.D)  CAN4 TX (T.B.D)  CAN5 RX (T.B.D)  CAN5 TX (T.B.D)  CAN6 RX (T.B.D)  CAN6 TX (T.B.D)  CAN7 RX (T.B.D)  CAN7 TX (T.B.D) |
| RLIN3n | RLIN3n unit | 0 -> 3 | 0 -> 7 | - |
| Select baud rate | 300  600  1200  2400  4800  9600  19200  31250  38400  76800  153600  312500 | 300  600  1200  2400  4800  9600  19200  31250  38400  76800  153600  312500 | - |
| TAUD | Slave channel | 1-> 15 | 1 -> 15 | - |
| Master channel | 0, 2, 4, 6, 8, 10, 12, 14 | 0, 2, 4, 6, 8, 10, 12, 14 | - |
| Port name in SC (correspond for each channel) | P10\_0  P0\_0  P10\_1  P0\_1  P10\_2  P0\_2  P10\_3  P0\_3  P10\_4  P10\_8  P10\_5  P10\_9  P10\_6  P10\_10  P10\_7  P9\_0 (channel 0) | P10\_0  P0\_0  P10\_1  P0\_1  P10\_2  P0\_2  P10\_3  P0\_3  P10\_4  P10\_8  P10\_5  P10\_9  P10\_6  P10\_10  P10\_7  P9\_0 (channel 0) | - |

**Remarks** 1.For RH850/F1KM-S4, there are 34 channels for ADC unit 0 while ADC unit 1 includes 36 channels

2. For RS-CANFD for RH850/U2C, currently, in HWM of U2C, the target port have not decided yet.

### 3.2.1 ADC peripheral

The following describes the main features of ADC peripheral.

* Get the data from MATLAB to ADC port on model.
* Return the same value from MATLAB to user algorithms.

A picture containing text

Description automatically generated

**Figure 3‑1 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.

Graphical user interface, text, application, email

Description automatically generated

**Figure 3‑2 User Interface of ADC S-Function block**

The User Interface of ADC S-Function block includes “ADC Port Name”, “Unit”, “Max 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 ADC port name, the parameters (Unit, Max Channel, Port ID) and the name of the S-Function block will be changed automatically based on the current ADC port name.

Graphical user interface, text, application

Description automatically generated

**Figure 3‑3 Changing port name of ADC S-Function block**

Graphical user interface, text, application, Teams

Description automatically generated

**Figure 3‑4** **The changing result of ADC S-Function block**

### 3.2.2 Port peripheral

The following describes about the main features of Port peripheral.

* **For Port\_In block:**
* Get data from MATLAB and send to Port in VLAB.
* Send the received value to the user algorithms.
* **For Port\_Out block:**
* Set the value from user algorithms to the target port name.
* Send the value of port to the MATLAB (output of the MATLAB Co-simulation).

A picture containing table

Description automatically generated A picture containing table

Description automatically generated

**Figure 3‑5 S-Function block of Port**

The following describes the User Interface of Port S-Function block that supports user selection and changes ports easier during model setting. The User Interface also has two types, which are corresponding to two types of Port S-Function blocks (Port\_In and Port\_Out).

Graphical user interface, text, application, email

Description automatically generated Graphical user interface, text, application, email

Description automatically generated

**Figure 3‑6 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.

A picture containing graphical user interface

Description automatically generated

**Figure 3‑7 Changing port name of Port S-Function block**

Graphical user interface, text, application, chat or text message, Teams

Description automatically generated

**Figure 3‑8 The changing result of Port S-Function block**

### 3.2.3 RS-CANFD peripheral

The following describes about 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).

Table

Description automatically generated Table

Description automatically generated

**Figure 3‑9 S-Function block of RS-CANFD**

The following describes the User Interface of RS-CANFD S-Function block that supports users selecting and changing RS-CANFD ports easier during the model set. The User Interface also has two types, which are corresponding to two RS-CANFD S-Function blocks (CAN\_Transmission and CAN\_Reception).

Graphical user interface, application

Description automatically generated

**Figure 3‑10 User Interface of RS-CANFD S-Function block**

The User Interface of RS-CANFD S-Function block includes “RS-CANFD unit”, “RS-CANFD channel”, “Port Name” parameters. The “RS-CANFD unit” and “RS-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: ETVPF prepared the CAN ID array (with the default ID message value as CAN unit number) for each CAN unit. Users can change it as the demand. ETVPF support get 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.

Graphical user interface, application

Description automatically generatedGraphical user interface, application, email

Description automatically generated

**Figure 3‑11 Information from CAN Pack and CAN Unpack**

- Model structure between MATLAB and VLAB:

Diagram

Description automatically generated

**Figure 3‑12 Model structure for using RS-CANFD**

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

- ETVPF support Vehicle Network Toolbox for integrating with CAN module of ETVPF.

### 3.2.4. RLIN3n peripheral

The following descriptions describe about the main features of RLIN3 peripheral.

* **For RLIN3n Transmission block:**
* This block receives the double data type from User Algorithm and transmit 8 bits through RLIN3 port. After that, it still receives data until it gets full data of 64 bits, then it transfers data to MATLAB (double value).
* **For RLIN3n Reception block:**
* This function receives the double data from MATLAB and transmit 8 bits through RLIN3 port. After that, it still receives data until it gets full data of 64 bits, then it transfers data to User Algorithm (double value).

Text

Description automatically generated with medium confidence Diagram

Description automatically generated with medium confidence

**Figure 3‑13 S-Function block of RLIN3n**

The following descriptions describe how the User Interface of RLIN3n S-function block is used to support user selection and change RLIN3n ports easier during the setting model. The User Interface includes two types, which are corresponding to two RLIN3n S-function blocks (RLIN3n\_Send and RLIN3n\_Receive).

Graphical user interface, text, application, email

Description automatically generated Graphical user interface, text, application, email

Description automatically generated

**Figure 3‑14 User Interface of RLIN3n S-Function block**

The User Interface of RLIN3 S-Function block includes “RLIN3 unit” and “Select baud rate” parameters.

The purpose of these parameters is to specify the target port name of RLIN3n 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 RLIN3n, the displaying name of S-function block will be changed automatically based on the current selecting RLIN3n unit.

Graphical user interface

Description automatically generated

**Figure 3‑15 The changing result of RLIN3n S-Function block**

### 3.2.5 TAUD peripheral

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

* This block receives three input Start/Stop (boolean), pulse cycle (double), duty value (array of double value) from User algorithm to control ouput of PWM signals.
* Ouput of this block is an array of data signals (master channel and slave channels). Slave channels select by User in SC.
* PWM output: generate PWM data signal then send data to MATLAB (boolean).
* Diagram

  Description automatically generatedTriangle PWM output: generate Triangle PWM data signal then send data to MATLAB (boolean).

**Figure 3‑16 S-Function block of TAUD**

Graphical user interface, text, application, email

Description automatically generatedThe following descriptions describe how the User Interface of TAUD S-function block is used to support user selection and change TAUD ports easier during the setting model.

**Figure 3‑17 User Interface of TAUD S-Function block**

The User Interface of TAUD S-Function block includes “TAUD unit”, “Mode”, “Master channel” parameters.

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

Graphical user interface, application

Description automatically generatedAfter changing the master channel of TAUD, the displaying name of S-function block will be changed automatically based on the current selecting TAUD master channel.

**Figure 3‑18 Changing port name of TAUD S-Function block**

* **Chart, waterfall chart

  Description automatically generatedModel structure of connection between User block with TAUD S-Function block:**

**Figure 3‑19 Model structure for using TAUD**

User must connect TAUD block same as above structure for conduct ETVPF with TAUD. To support setting for multiple channels and connection between MATLAB and VLAB, use Mux and Demux to connect from User algorithm to TAUD block.

To keep the dimension of input and output are same, Input duty array must include a dummy value for master at the end of input array.

If user want to NOT change the input value after started, user can connect inputs to constant values. The constant value is same as initial setting value.

Example: connect Duty and Pulse to constant number

Output of TAUD is an array of data signals. To get data from each channel, user must extract data from array to single data signal by using Demux to get each data.

Output arrays include signal of master channel and slave channels. Input for slave is also array.

**- Support Model in the Loop Simulation (MILS) for TAUD block:**

ETVPF create TAUD block for executing MILS. To use this block, user must replace the S-function TAUD to MILS TAUD block then execute MILS.

Input of MILS TAUD includes duty, pulse, and start/stop trigger. When running the model, the S-function “mils\_taud” inside TAUD MILS will set the duty input for “PulseWidth” and pulse input for “Period” of the pulse generator. Please match the input of MILS TAUD and the initial value of the pulse generator (“PulseWidth” and “Period”).

In MILS, user don’t need to use Mux and Demux block.

Each block support for each TAUD channel. To use multiple channels, user must connect to multiple blocks.

Chart

Description automatically generated

**Figure 3‑20 Model structure for using TAUD MILS**

## 3.3 Executing Simulator Processor in the Loop Simulation

The following describes how to generate a SPILS environment necessary for Simulator Processor in the Loop Simulation (hereafter referred to as SPILS) 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 contained the S-function blocks of peripherals.

Graphical user interface, application

Description automatically generated

**Figure 3‑21 The S-function blocks of peripherals are in ETVPF package**

The sample model is Power Window model (slexPowerWindowExample.slx) that is used for the following explanation with ET-VPF. The feature of peripherals (Port, ADC, RS-CANFD, RLIN3n, TAUD) 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 (Port, ADC, RS-CANFD, RLIN3n, TAUD) must be in the layers under the Code generation target. If they are outside the Code generation target, the error can occur.

Graphical user interface

Description automatically generated

**Figure 3‑22 The Code generation target and S-Function blocks of peripherals**

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

**Table 3‑3 Code generation target of sample model**

|  |  |  |
| --- | --- | --- |
| **Sample model name** | **Code generation target** | **Block type** |
| slexPowerWindowExample.slx | Target\_Block\_ET\_VPF | Subsystem block |

**Table 3‑4 Peripherals under Code generation target**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Code generation target** | **Peripheral** | **Peripheral port name** | **Block name** | **Block type** |
| Target\_Block\_ET\_VPF | ADC | ANI<Unit><Port ID>\*1  AN<Unit><Port ID>\*2 | ADC\_<Number> | S-Function block |
| Port | P<Port>\_<Pin> | Port\_In\_<Number> | S-Function block |
| Port\_Out\_<Number> | S-Function block |
| RS-CANFD | CAN<Unit>\_TX | CAN\_Transmission \_<Number> | S-Function block |
| CAN<Unit>\_RX | CAN\_Reception \_<Number> |
| RLIN3n | RLIN3<RLIN3 unit>\_Send | RLIN3n\_Transmission\_<Number> | S-Function block |
| RLIN3<RLIN3 unit>\_Receive | RLIN3n\_Reception\_<Number> | S-Function block |
| TAUD | TAUD<Unit>\_<master channel> | TAUD\_<Number> | S-Function block |

\*1… Use for RH850/F1KM-S1 and RH850/F1KM-S4 device series.

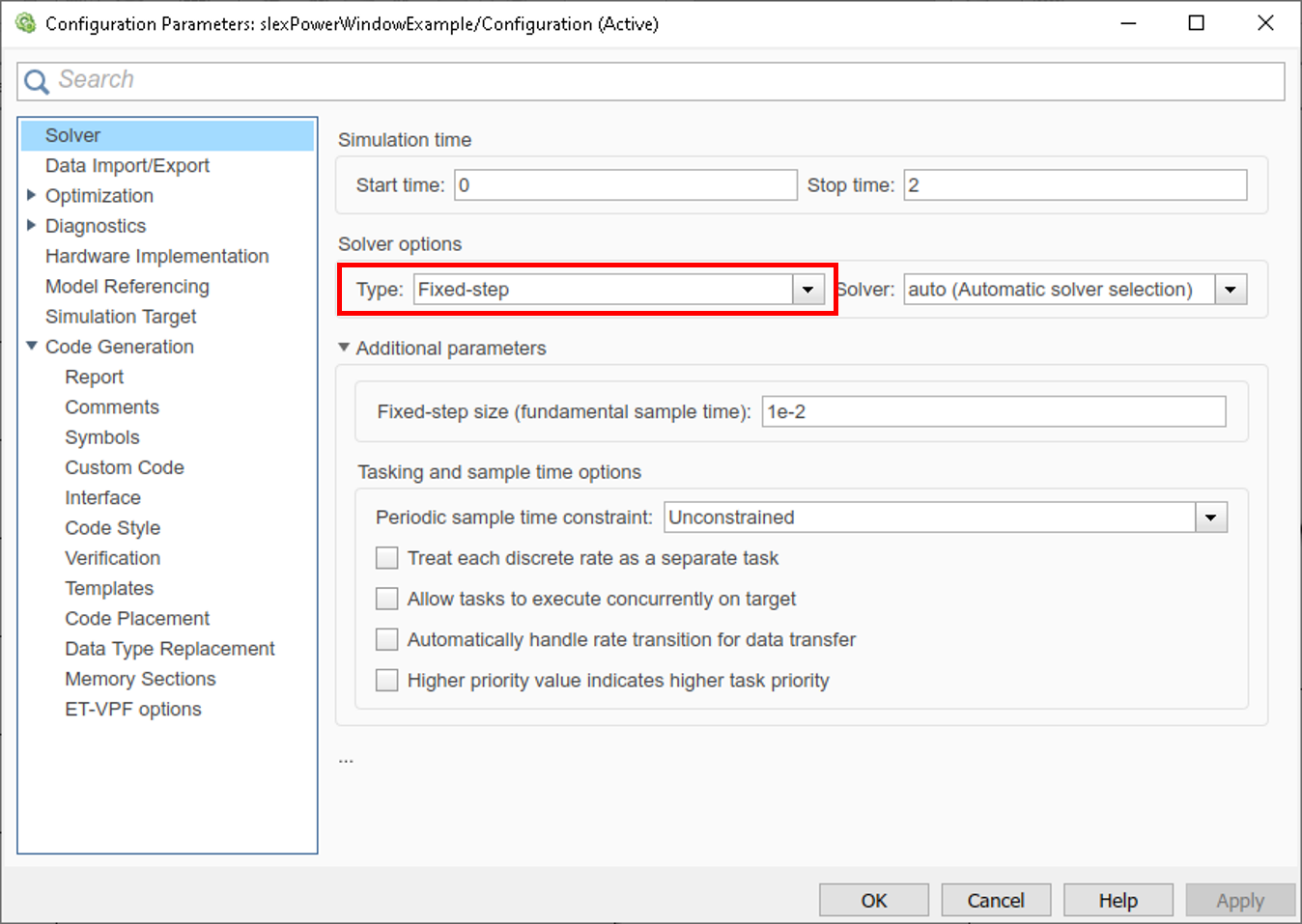
\*2… Use for RH850/U2C device series.

### 3.3.2 Setting configuration parameters

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

1. Open MATLAB R2017b or R2021a.
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 3‑23 [Solver] settings**

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

Graphical user interface, text, application, email

Description automatically generated

**Figure 3‑24 [Code Generation] settings**

* **Step 3:** In the [ET-VPF options], select necessary settings that described in the **Table 3‑5 ET-VPF Options**.

Graphical user interface, text, application, email

Description automatically generated

**Figure 3‑25 [ET-VPF options] settings**

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

**Table 3‑5 ET-VPF Options**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item name** | **Description** | | | |
| Cygwin Installation Directory \*1 \*15 | Specifies the folder where Cygwin has been installed (the folder where bin/bash.exe is stored) as an absolute path. | | | |
| [Use default Cygwin Installation Directory] checkbox | Specifies the default folder where Cygwin has been installed. It is “C:/cygwin64”. | | | |
| [Select Cygwin Installation Directory] button \*1 \*2 | Clicking this button displays the dialog box for selecting the absolute path of the folder where the Cygwin is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [Cygwin Installation Directory] field. | | | |
| VLAB Installation Directory \*3 \*15 | Specifies the folder where VLAB has been installed (the folder where vlab-ide.exe is stored) as an absolute path. | | | |
| [Use default VLAB Installation Directory] checkbox | Specifies the default folder where VLAB has been installed. It is “C:/Program Files/Vlab”. | | | |
| [Select VLAB Installation Directory] button \*3 \*4 | Clicking this button displays the dialog box for selecting the absolute path of the folder where the VLAB is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [VLAB Installation Directory] field. | | | |
| SC Installation Directory \*5 \*15 | Specifies the folder where Smart Configurator has been installed (the folder where SmartConfigurator.exe is stored) as an absolute path. | | | |
| [Use default Smart Configurator Installation Directory] checkbox | Specifies the default folder where Smart Configurator has been installed. It is “C:/Program Files (x86)/Renesas Electronics/SmartConfigurator/RH850/eclipse”. | | | |
| [Select Smart Configurator Installation Directory] button \*5 \*6 | Clicking this button displays the dialog box for selecting the absolute path of the folder where the Smart Configurator is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [SC Installation Directory] field. | | | |
| IDE Mode | Selects the type of project file that is loaded when the VLAB starts and whether or not a series of processing including the download of a load module is performed after the VLAB start-up. | | | |
| Create Project (default) | The default project file provided by ET-VPF is loaded. | | |
| Build Tool \*7 | Selects the Build tool for the generated project, this indicates the compiler will be used to generate the load module. | | | |
| Renesas Compiler \*8 | | Selects any of Renesas compilers, which will be determined by CS+ is reflected in the [CS+ Installation Directory] field. | |
| CS+ Installation Directory \*9 \*15 | Specifies the folder where CS+ has been installed (the folder where CubeSuiteW+.exe is stored) as an absolute path. | | | |
| [Use default CS+ Installation Directory] checkbox | Specifies the default folder where CS+ has been installed. | | | |
| [Select CS+ Installation Directory] button (\*9 \*10) | Clicking this button displays the dialog box for selecting the absolute path of the folder where the CS+ is installed. Folder specifications made in the dialog box that is opened by this button are reflected in the [CS+ Installation Directory] field. | | | |
| Device Series \*11 | Selects the series name of the microcontroller being used. | | | |
| <Device Series Name> \*12 | | | 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. |
| OSTM PCLK \*13 | The OS Timer value corresponds to each device series. | | | |
| [Check Available License] button \*14 | Displays list of available requiring licenses in ET-VPF System. | | | |
| [About] button | Displays version information and copyright information of ET-VPF. | | | |

\*1… When Cygwin has not been installed in the folder specified with the dialog box (bin/bash.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 [Cygwin Installation Directory].

This setting is only valid if “Create Project” is selected for [IDE Mode], the [Use default Cygwin Installation Directory] checkbox is not checked.

\*2… When the [Use default Cygwin Installation Directory] checkbox is checked if the [Select Cygwin Installation Directory] button is clicked, an error message displays.

\*3… When VLAB has not been installed in the folder specified with the dialog box (vlab-ide.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 [VLAB Installation Directory].

This setting is only valid if “Create Project” is selected for [IDE Mode], the [Use default VLAB Installation Directory] checkbox is not checked.

\*4… When the [Use default VLAB Installation Directory] checkbox is checked, if the [Select VLAB Installation Directory] button is clicked, an error message displays.

\*5… When Smart Configurator has not been installed in the folder specified with the dialog box (SmartConfigurator.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 [SC Installation Directory].

This setting is only valid if “Create Project” is selected for [IDE Mode], the [Use default Smart Configurator Installation Directory] checkbox is not checked.

Remark This setting is unavailable if using RH850/U2C device series.

\*6… When the [Use default Smart Configurator Installation Directory] checkbox is checked if the [Select Smart Configurator Installation Directory] button is clicked, an error message displays.

Remark This setting is unavailable if using RH850/U2C device series.

\*7… This setting is only valid if “Create Project” is selected for [IDE Mode].

\*8… When [Build Tool] is set to “Renesas Compiler”, the build tool is decided by CS+ at the source code compiling time.

\*9… When CS+ has not been installed in the folder specified with the dialog box (CubeSuiteW+.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 [CS+ Installation Directory].

This setting is only valid if “Create Project” is selected for [IDE Mode], the [Use default CS+ Installation Directory] checkbox is not checked.

\*10… When the [Use default IDE Install Directory] checkbox is checked, if the [Select IDE Install Directory] is clicked, an error message displays.

\*11… This setting is only valid if “Create Project” is selected for [IDE Mode], the “Embedded Target for Virtual Platform RH850” license is valid to use.

\*12… The list of supported devices in the current ET-VPF version (refer to **Table 1‑1 Supported devices**).

\*13… When “Renesas” and “RH850/F1KM-S1” are selected, the [OSTM PCLK] must be set to “4000000”. This value is also set as the default value.

When “Renesas” and “RH850/F1KM-S4” are selected, the [OSTM PCLK] must be set to “2000000”. This value is also set as the default value.

When “Renesas” and “RH850/U2C” are selected, the [OSTM PCLK] must be set to “80000000”. This value is also set as the default value.

\*14… When clicking on this button, the dialog box displays and shows the list of required licenses. These can be used freely.

\*15… The path of CS+, Cygwin, Smart Configurator and VLAB is only support the special characters that described in the following table.

**Table 3‑6 The supported special characters**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Special Characters** | **CS+ and Cygwin** | **Smart Configurator** | **VLAB** | **<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 | O | O |
| , | - | - | - | - |
| . | O | O | O | - |
| ‘ | - | - | - | - |
| ‘’ | - | O | O | - |
| ; | - | - | - | - |
| space | - | O | O | - |

*\** 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 SPILS environment

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

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 SPILS environment.

**Table 3‑7 Provided command**

|  |  |
| --- | --- |
| **Command name** | **Description** |
| run\_vlab | Generates a SPILS environment and executes SPILS automatically |

1. Select the Code generation target on the model.
2. Execute generation of SPILS 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\_vlab [Enter]

1. After executing “**run\_vlab**” command, the SC\_project folder will be created. And a message box is displayed to notify for the user about the full path of SC\_project.

Remark If using RH850/U2C device series, this message box is not displayed and do not execute the feature which described in section **3.3.3.1 Generating the peripherals’ source code by SC**.

Graphical user interface, application, Teams

Description automatically generated

**Figure 3‑26 Message box notify the full path of SC\_project**

#### 3.3.3.1 Generating the peripherals’ source code by SC

1. After that, the SC Window will be displayed. Users can configure detailed settings and generate peripherals’ source code as below:

* **Step 1:** Select [New Configuration File] or [File] -> [New…] to create new SC project.

Graphical user interface, application, Word

Description automatically generated

**Figure 3‑27 Creating new SC project**

* **Step 2:** Select platform and toolchain settings:
* Select Device name in [Device] setting **(1)** (***Note:*** Currently, only F1KM device series are supported).
* Select Toolchain type in [Toolchain] setting **(2)** (***Note:*** Currently, only Renesas toolchains are supported).
* Fill [File name] and select [Location] settings **(3)** (***Note:*** The [Location] must be the full path of the SC project displayed in the message box).
* Click [Finish] button **(4)**.

Graphical user interface, text, application

Description automatically generated

**Figure 3‑28 Selecting SC settings**

* **Step 3:** Import available XML file to SC. (Skip this step if model contains only CAN)
* The XML file of each device name is provided to support the user in importing the components easier. This XML file (with the name “newxml.xml”) will be generated in the working directory.
* To import available XML file: Select [Component] **(1)**, select [Import Configuration] **(2)** settings.

Graphical user interface, application

Description automatically generated

**Figure 3‑29 Importing Configuration**

* Select [Import File] **(1)** and select the available XML file (newxml.xml).
* Click [Select All] button **(2)** if all configurations are necessary (***Note:*** Only select the necessary configurations).
* Click [Next >] button **(3)**.

Graphical user interface, text, application

Description automatically generated

**Figure 3‑30 Importing available XML file**

* The configuration of all peripherals is prepared. Click [Finish] button to complete importing configuration.

A computer screen capture

Description automatically generated with low confidence

**Figure 3‑31 Completed importing configuration**

Remark 1. ET-VPF also supports execution configuration functions of peripherals automatically. To do this, the configuration function name must be a determined format. Therefore, when generating peripherals’ source code by SC, the configuration name must be the default name.

2. For each RLIN3n unit, it only has one configuration function, which corresponds to UART resource (e.g., RLIN30 corresponds to “UART0” resource). If we use two RLIN3n units with the same UART resource, the error will occur.

3. For TAUD, user must select the clock source and slave channels in SC GUI.

* **Step 4:** Change detail settings (skip this action if model contains only CAN) and generate source code for peripherals.

Graphical user interface, application

Description automatically generated

**Figure 3‑32** **Setting slave channels and clock source for TAUD**

Graphical user interface, application

Description automatically generated

**Figure 3‑33 Selecting detail settings and generating peripherals’ source code**

Remark When changing detail settings, these settings must be the same as the peripherals’ settings on the model. If these settings are different with the peripherals’ settings on model, it causes SPIL Simulation incorrect.

* **Step 5:** Close SC Window.

1. The source code of peripherals will be generated under the SC\_project folder.

Graphical user interface, text, application

Description automatically generated

**Figure 3‑34 The peripherals’ source code generated under SC\_project**

#### 3.3.3.2 Generating the target SPILS environment

1. The target SPILS environment will be generated in the working directory (the location which contains the target model).

The target SPILS environment includes “slprj” and “<Code generation target>\_etvpf” folders. And the necessary files will be generated in these folders.

* Source files of target device.
* C source files.
* Make file
* VLAB execution file
* Peripherals’ wrapper source files
* Python source files.
* Define.h, OSTM\_define.h, target.out.

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 PIL sequential execution (with block name is “VLAB Bridge”) for the model file to be copied.

Graphical user interface, application

Description automatically generated

**Figure 3‑35 Example of replacing to the block for PIL sequential execution**

### 3.3.4 Compiling generated source code

1. After that, the C source files, and source files of the target device (with the extension is \*.c, \*.asm) will be compiled to object files (with the extension is \*.o) via Cygwin.

The information related to the compiling will be displayed on MATLAB Command Window.

Table

Description automatically generated

**Figure 3‑36 Compiling is displayed on MATLAB Command Window**

1. Then, the “target.out” file is also generated. After that, the object files and the “target.out” file are also stored in the working directory, under “<Code generation target>\_etvpf” folder.

### 3.3.5 Executing SPILS

1. After the compiling is completed, the VLAB is opened. The RH850 Virtual Platform of RH850/F1KM, RH850/U2C toolbox, “target.out” and source files in the working directory are loaded.

The SPILS execution starts on both the Simulink model and VLAB.

Graphical user interface, diagram

Description automatically generated

**Figure 3‑37 The SPILS execution**

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

Graphical user interface

Description automatically generated

**Figure 3‑38 The result of scope**

Remark Please ignore the first step. Because, at that time, MATLAB and VLAB are connected and a dummy value is sent to MATLAB by the MATLAB Co-simulation.

## 3.4 Time measurement

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

The Send and Receive progress from both Simulink model and VLAB depend on the Sampling Time.

Diagram

Description automatically generated

**Figure 3‑39 Time measurement method**

### 3.4.1. Structure of Simulink Model for measurement

Graphical user interface

Description automatically generated

**Figure 3‑40 Structure of Simulink model for measurement**

Remarks (\*1): The measured block meet all requirement for measurement.

(\*2): User decision: Not measure this block. So, they don't need to transform it to Atomic Subsystem. Refer to section **3.4.3. How to change Normal Subsystem to Atomic Subsystem**.

(\*3): Although it is Atomic Subsystem, but user do not define 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 **3.4.2. Input file for Measurement**.

3. It must be the Atomic Subsystem.

### 3.4.2. 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:

Text

Description automatically generated

**Figure 3‑41 Examples of input\_subsystem.txt for Time Measurement**

### 3.4.3. 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”.

Graphical user interface, text, application, email

Description automatically generated Graphical user interface, text, application, email

Description automatically generated

**Figure 3‑42 Change Normal Subsystem to Atomic Subsystem**

### 3.4.4 Graph Viewer

#### 3.4.4.1 Input data

Execution time dictionary:

“Subsystem1”: [start end start end start end ….]

“Subsystem2”: [start end start end start end ….]

“Subsystem3”: [start end start end start end ….]

By using this method, can get/set start and end time by using the function label.

Input 1: execution\_data.csv file refer to upper data structure.

* Its unit is picosecond.
* It contains the execution time of each step and start end of each subsystem.
* The value of start end is relative value.

Graphical user interface, application, table, Excel

Description automatically generated

Input 2: input\_subsystem.txt file refer to section **3.4.2. Input file for Measurement**.

#### 3.4.4.2 Output figure

The following is the output figure that will be displayed when executing the 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]

Chart

Description automatically generated

**Figure 3‑43 Output figure of time measurement**

Remark Currently, in 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 show the execution time of each Subsystem for one step or more steps belonging to the user choose in “Start Step” and “End Steps”.

Graphical user interface

Description automatically generated

**Figure 3‑44 Press [Show Step Graph] to show execution time each Subsystem**

Then set “Start Step” and “End Start” (e.g., Start Step = 1, End Step = 5).

Table

Description automatically generated

**Figure 3‑45 Example of Execution Time Detail at step 1**

1. Change the scale of the step by pressing [Change Scale] button then set “Start Step” and “End Start” (e.g., Start Step = 1, End Step = 228).

Chart

Description automatically generated

**Figure 3‑46 Press [Change Scale] to change scale of step**

Graphical user interface, application

Description automatically generated

**Figure 3‑47 Set “Start Step” and “End Step” value**

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

1. Press [MAX/min] to show which steps display the maximum execution time and minimum execution time.

Chart

Description automatically generated with medium confidence

**Figure 3‑48 Show Max/Min execution time**

Then the Max Min Dialog shows the step has execution time maximum/minimum and the value of execution time of this step.

Press [Show step graph] in Maximum/Minimum Section to show execution time detail of this step.

Graphical user interface, text, application

Description automatically generated

**Figure 3‑49 Max/Min Dialog**

# 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. In the current SC (version 1.5.0), it has still not supported RH850/F1KM-S4 - 272 pins device series yet. Consequently, this device can not be used in this development yet.
2. The name of measured blocks must be unique.

Currently, the method to get the generated function for the measured block is using the information in the comment. Its comment only has the Subsystem name. Then, it cannot identify the target measured block by the full Subsystem path.

1. ~~For RS-CANFD~~
2. ~~This version does not support s~~~~ome functions: the remote frame, extend ID and receiving filter.~~
3. ~~The data size, which is used for sending and receiving are fixed to 8 bytes (double value).~~
4. ~~The message ID currently is fixed to the CAN unit number.~~

~~These limitations would be updated in the next version.~~

1. ~~It only supports CAN TOKEN Port:~~

~~In the VLAB environment, ETVPF uses the TOKEN Port, which is virtual CAN port of VLAB (For example: CAN0RX is RSCAN0\_TOKEN\_RX\_DATA0 and RH850.RSCAN0\_TOKEN\_RX\_CTRL0) to connect to the CAN Bus to maintain the simulation speed.~~

~~ETVPF does not use the CAN Port I/O for connecting between CAN module (CAN TX and CAN RX) and the CAN Bus. Therefore, when using the real device, the user must map the CAN module to the Port I/O before connecting to the CAN Bus.Support only CAN TOKEN Port:~~

~~Due to the use of the TOKEN Port, ETVPF will not guarantee that the behavior of using the Port I/O is completely correct compared to a real device.~~

1. There are some functions of driver code that are not guaranteed:

* R\_Config\_<CAN Channel>\_Transmission\_Stop function: This function is not compatible with ET-VPF. Therefore, this function is just for user reference and not guaranteed.
* CAN\_Common\_Reception\_Init function: for executing SPILS, ET-VPF uses the CAN\_Common\_Init function to initialize for both transmission and reception. Therefore, this function is just for user reference and not guaranteed.

1. The features for RH850/U2C are preliminary because it's under development.
2. For ADC in RH850/U2C, this version only supports for Unit 1 (ADCK1), Scan group 1, Multicycle scan mode.

For ADC in RH850/F1KM, this version only supports for Scan group 1, Multicycle scan mode.

1. pin assigns P27 group (P27\_0 only)
2. To synchronize the communication between VLAB and MATLAB, sampling time must higher than the execution time of algorithm each step. When the execution time higher than sampling time, the output is not guaranteed. For specific execution time, it describes in the “Time” collumn of execution\_data.csv in “<Code generation target>\_etvpf” folder.
3. Duty time and pulse time of TAUD must be higher than sampling time.

As ETVPF spec, signal of PWM Output will be sent to MATLAB each sampling time

For TAUD, when using duty time and pulse time smaller than sampling time, during receive output of PWM from VPF to MATLAB, some signal may be missed. Thus, Duty time and pulse time of TAUD must be higher than sampling time.

1. TAUD output signal of VPF delays a little bit

In actual VPF, the TAUD clock has some delay at initialization and it may be the same as the actual chip (not ideal as spec, high value immediately when start TAUD).

For workaround, we prepared a unit delay block at the output of MILS TAUD block. By applied this method, waveform of MILS and SPILS are the same, except the first step value we not sure (currently the initial value of unit delay is 0)

1. TAUD MILS for Triangle PWM Output is not supported.
2. For CA mode, user can enable it, but it is just the evaluation feature in this version

If user wants to use CA mode, please follow the below steps:

**Step 1**: Run ETVPF for the first time and keep the environment.

**Step 2**: Change some strings in ETVPF\_testbench\_sample.py to replace FastIss to CAIss.

* vlab.read\_register("RH850.CPUSS.PE1\_FastIss.r31") change to vlab.read\_register("RH850.CPUSS.PE1\_CAIss.r31")
* vlab.load('rh850.f1km', ['--device-type=S1', '-t', './sim.py']) change to vlab.load('rh850.f1km', ['--device-type=S1', '-t', './sim.py', '-iss=ca'])

**Step 3**: Rerun the current Simulink model manually

**Step 4**: Rerun the ETVPF\_testbench\_sample.py manually.

1. The installation path of ET-VPF package, CS+, Cygwin, Smart Configurator and VLAB cannot contain some special characters (refer to **Table 3‑6 The supported special characters** for more detail).

## 4.2 Simulink models

### 4.2.1 Available Strings for Paths and Block Name

Do not use 2-byte characters (Japanese, etc.), spaces, slashes, line feeds, or hyphens to name for 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 that given by MATLAB. 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 the complex number data is not supported.

## 4.3 Construction and Simulation

### 4.3.1. Length of Path to Code Generation Folder

The length of path character to files or directories is restricted in the Windows platform (limit of 260 characters). If the length of path character 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) exceed 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 is in sleeping or hibernating mode during 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 exists 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 that the ET-VPF installation drive and the working drive that stores the model are the same. Otherwise, an error will occur.

# 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 SPIL simulation is running.

Remark Error messages output by ET-VPF is not linked to VLAB. 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 Box

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] | E0101  The Cygwin directory setting is incorrect. |
| --- | --- |
| [Explanation] | This error message is displayed when information on the correct installation destination has not been set in [Cygwin Path]. |
| [Action by User] | 1. Set the correct path of the installation destination of Cygwin to [Cygwin Path].  The folder containing “/bin/bash.exe” must be specified. 2. Check that Cygwin has normally been installed. |
| [Message] | E0102  The VLAB directory setting is incorrect. |
| [Explanation] | This error message is displayed when information on the correct installation destination has not been set in [VLAB Path]. |
| [Action by User] | 1. Set the correct path of the installation destination of VLAB to [VLAB Path].  The folder containing “vlab-ide.exe” must be specified. 2. Check that VLAB has normally been installed. |
| [Message] | E0103  The CS+ directory setting is incorrect. |
| [Explanation] | This error message is displayed when information on the correct installation destination has not been set in [CS+ Path]. |
| [Action by User] | 1. Set the correct path of the installation destination of CS+ to [CS+ Path].  The folder containing “CubeSuiteW+.exe” must be specified. 2. Check that CS+ has normally been installed. |
| [Message] | E0104  The Smart Configurator directory setting is incorrect. |
| [Explanation] | This error message is displayed when information on the correct installation destination has not been set in [SC Path]. |
| [Action by User] | 1. Set the correct path of the installation destination of Smart Configurator to [SC Path].  The folder containing “SmartConfigurator.exe” must be specified. 2. Check that Smart Configurator has normally been installed. |
| [Message] | E0106  The default directory is selected as the current installation directory.  Deselect the use of the default directory. |
| [Explanation] | 1. This error message is displayed when [Select Cygwin Installation Directory] button is clicked, while the [Use default Cygwin Installation Directory] checkbox is checked. 2. This error message is displayed when [Select VLAB Installation Directory] button is clicked, while the [Use default VLAB Installation Directory] checkbox is checked. 3. This error message is displayed when [Select Smart Configurator Installation Directory] button is clicked, while the [Use default Smart Configurator Installation Directory] checkbox is checked. 4. This error message is displayed when [Select CS+ Installation Directory] button is clicked, while the [Use default CS+ Installation Directory] checkbox is checked. |
| [Action by User] | 1. Uncheck [Use default Cygwin Installation Directory] checkbox. 2. Uncheck [Use default VLAB Installation Directory] checkbox. 3. Uncheck [Use default Smart Configurator Installation Directory] checkbox. 4. Uncheck [Use default CS+ Installation Directory] checkbox. |
| [Message] | E0110  A license is not registered. |
| [Explanation] | This message displays when no license or license was expired on your system. |
| [Action by User] | 1. 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 RH850 device series is invalid. |
| [Action by User] | 1. If you don’t have the license for RH850 device series, register with Renesas Electronics.   If you have got license RH850 device series, check if it is put in the ET-VPF installation. To confirm the availability of the license, please [Check Available License] on [ET-VPF options] panel. |

## 5.3 Errors during SPILS execution

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

**Table 5‑2 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 components are correct. 2. If the above information is correct, re-execute SPILS to generate the execution time measurement result again. |

# CHANGE HISTORY

The main changes from the previous versions of ET-VPF.

|  |  |  |
| --- | --- | --- |
| **Version** | **Category** | **Summary** |
| E1.00.00 | Environment | Initialize the ET-VPF System Specification.  Supporting ET-VPF for F1KM alpha version. |
| License policy and license management model |
| S-Function block of peripherals |
| Configuration dialog |
| SPIL Simulation Execution |
| Time measurement |
| Points for caution |
| Error messages |
| V1.00.00 | Environment | Update the contents for RLIN3n  Update the contents of F1KM-S4  Update the contents of U2C alpha version  Update the contents of TAUD  Update the contents of RS-CANFD for F1KM and U2C Device Series.  Update the contents of ET-VPF installer  Update the contents of ADC unit 1  Update the S-function blocks to MATLAB Simulink Library Browser |
| Target Device Series |
| License policy and license management model |
| S-Function block of peripherals |
| Generating the peripherals’ source code by SC |
| Time measurement |
| Points for caution |
| Embedded sample model |
| Installation |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Changed (Revised) History (ET-VPF E1.00.00)** | | | | | | |
| No. | \* 1 change notes | | | \* 2  Creation date  Designation date | \* 3  Pulling up  Technical document | Author |
| Content | Page | Reason |
| 1 | Initialize the document  Update the information related to F1KM alpha version (Req. 01) | All | Supporting ET-VPF for F1KM alpha version | Feb 21, 2022 | None | Hong Tieu  Lam Nguyen |
| 2 | Update the unclear, incorrect contents. | All | Fix action items after PR1. | Mar 02, 2022 | None | Hong Tieu  Lam Nguyen |
| 3 | Add information related to RS-CANFD | 10,11,17 | Supporting RS-CANFD for ET-VPF | Mar 02, 2022 | None | Son Thai |
| 4 | Add information related to REL’s additional requests | 33, 55 | Supporting REL’s additional requests | Mar 07, 2022 | None | Hong Tieu |
| \* 1. Briefly describe the content and reasons for the revision. Also, if there is a document describing the reason/ contents, write that number.  \* 2. Upper row: Creation month/ day: Lower row: Enter the designation date (especially when the specified date is required). Applicable designation date Applicable from the date of receipt if not stated.  \* 3. If there is a technical document (including IMD) that becomes unnecessary except the old version after the designated designation date, enter that number (including version number). | | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Changed (Revised) History (ET-VPF V1.00.00)** | | | | | | |
| No. | \* 1 change notes | | | \* 2  Creation date  Designation date | \* 3  Pulling up  Technical document | Author |
| Content | Page | Reason |
| 1 | Update the contents that related to F1KM-S4 | 1, 35, 57 | Supporting ET-VPF for F1KM-S4 | June 27, 2022 | None | Hong Tieu  Giang Nguyen |
| 2 | Update the contents that related to RLIN3n | 1, 15, 17, 25, 30, 40, 57 | Supporting RLIN3n for ET-VPF | Jun 27, 2022 | None | Hong Tieu  Giang Nguyen |
| 3 | Update the contents that related to U2C | 1, 7, 17, 8, 9, 16,17, 30, 35, 37, 43, 52, 57 | Supporting ET-VPF for U2C alpha version  Supporting Port, ADC for U2C | Jul 01, 2022 | None | Hong Tieu  Son Thai |
| 4 | Update the contents that related to TAUD | 1, 15, 17, 26, 29, 30, 57 | Support TAUD for ET-VPF | Jul 14, 2022 | None | Tinh Le |
| 5 | Update the contents that related to CAN | 16, 23,23,24,52 | Support unsupported features of CAN in F1KM E1.00 | Jul 13, 2022 | None | Phuc Giang |
| 6 | Update the contents for F1KM-S4  Update the contents for U2C | 9, 17 | Update after implement coding phase | Aug 22, 2022 | None | Son Thai |
| 7 | Update contents after fixed bugs | 27, 52 | Update contents after fixed bugs | Sep 12, 2022 | None | Tinh Le  Phuc Giang |
| 8 | Update contents related to SC for CAN | 38, 40 | Update the contents after fixed REL’s comments | Sep 19, 2022 | None | Phuc Giang |
| 9 | Update the contents of U2C alpha version after received new toolbox | 6, 13, 29, 46 | Update the contents of U2C alpha version after received new toolbox | Sep 22, 2022 | None | Son Thai |
| 10 | Update the contents for installation and uninstallation | 9, 11, 13, 14, 57 | Update the contents for ET-VPF installer | Oct 17, 2022 | None | Hong Tieu |
| 11 | Update the installation procedure | 11, 13, 14 | Update the contents for ET-VPF installer | Oct 31, 2022 | None | Hong Tieu |
| 12 | Update ADC unit 1 for RH850/F1KM-S4 | 2, 7, 17, 18, 57 | Support ADC unit 1 for RH850/F1KM-S4 | Nov 02, 2022 | None | Giang Nguyen |
| 13 | Update the list of special characters in path  Update the limitations | 8, 16, 35, 52, 53 | Update for REL’s comments | Nov 25, 2022 | None | Hong Tieu |
| 14 | Update peripheral S-function blocks to MATLAB Simulink Library Browser | 2, 12, 15, 16, 29, 57 | Preparing peripheral S-function blocks to MATLAB Simulink Library Browser | Nov 25, 2022 | None | Giang Nguyen |
| 15 | Add supporting CAN for U2C information | 17 | Support CAN for U2C | Nov 29, 2022 | None | Phuc Giang |
| 16 | Add MATLAB 2021a content | 7, 31 | Support MATLAB R2021a | Dec 02, 2022 | None | Tinh Le |
| 17 | Update content in the section 1.5 | 9 | Update the file list of Renesas Electronics MCU Tools | Dec 06, 2022 | None | Hong Tieu |
| 18 | Update for Time Measurement method | 45 | Update time measurement method in ETVPF | Dec 19, 2022 | None | Son Thai |
| \* 1. Briefly describe the content and reasons for the revision. Also, if there is a document describing the reason/ contents, write that number.  \* 2. Upper row: Creation month/ day: Lower row: Enter the designation date (especially when the specified date is required). Applicable designation date Applicable from the date of receipt if not stated.  \* 3. If there is a technical document (including IMD) that becomes unnecessary except the old version after the designated designation date, enter that number (including version number). | | | | | | |

End of Document