Project Conversion between e^2 studio and CS+, Notes and Tips

Renesas Electronics Corporation
IoT and Infrastructure Business Unit
Software Development Division
February 21, 2022

R20UT3239EJ0300

AGENDA

- Purpose of This Document **Page 03**
- How to Convert Projects **Page 04**
- Restrictions **Page 09**
- Conversion of Build Variables and Placeholders **Page 22**

Purpose of This Document

Existing projects can be converted between e^2 studio and CS+ through the project import and export functions although there are some restrictions due to the differences in project management policies between the tools.

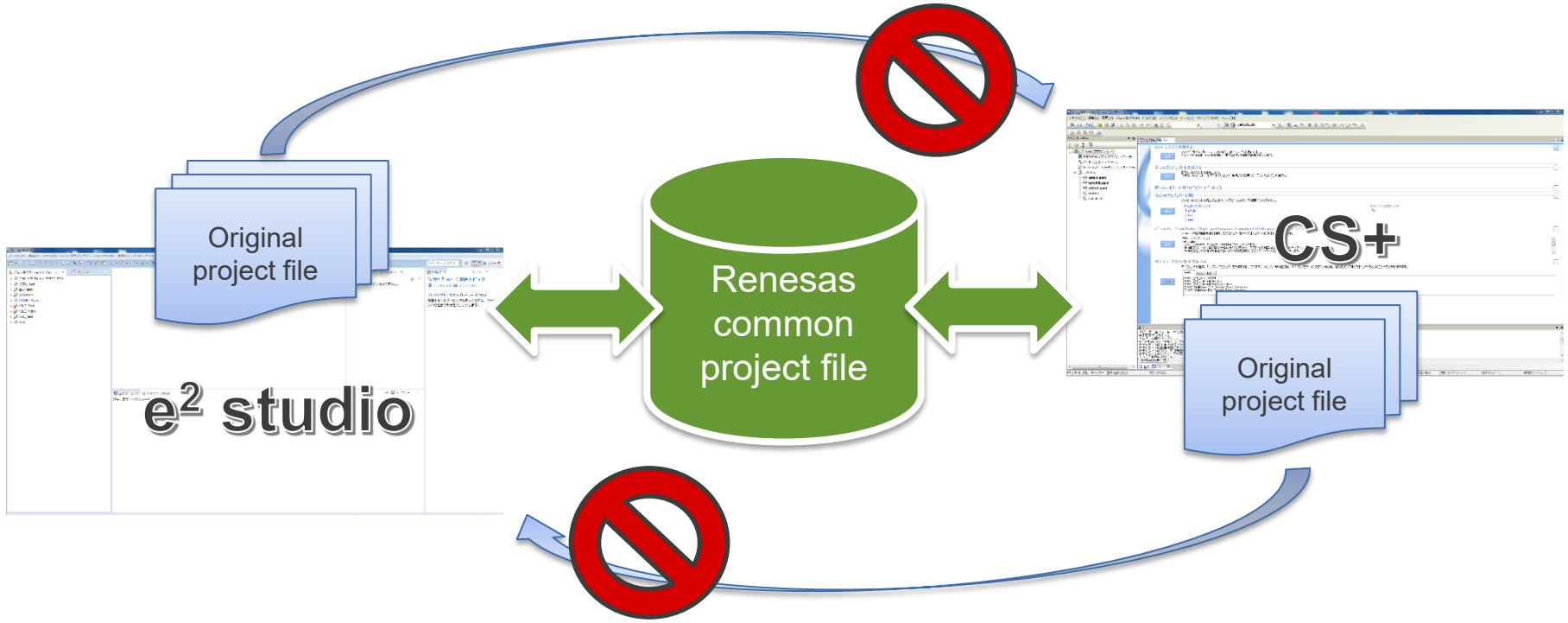
This document describes each restriction on functions that should be noted during project conversion when using e^2 studio 2022-01 and CS+ V8.07.00.

Note: The tool name "CubeSuite+" was changed to "CS+" from V3.00.00 released on Oct. 1, 2014.
The descriptions in this document apply to both CS+ and CubeSuite+ unless otherwise noted.

How to Convert Projects

What is Project Conversion?

The format for saving project files differs between e^2 studio and CS+. Therefore, a project created in either of these tools cannot be used in the other tool. To solve this problem, each tool now supports a new function for outputting project files in a Renesas common format. Using Renesas common project files enables bidirectional project conversion.



Project Conversion Procedures

A project can be converted between CS+ and e^2 studio with the following procedures.

- CS+ projects into e^2 studio (CS+ => e^2 studio)
  CS+ automatically generates "Renesas common project files (*.rcpe)".
  Use these files to convert a project.

  1. Select menu [File] > [Import...] in e^2 studio and specify the “Renesas common project files (*.rcpe)” output by CS+.
  2. The imported projects can be used in e^2 studio.

- e^2 studio projects into CS+ (e^2 studio => CS+)
  e^2 studio automatically generate "Renesas common project files (*.rcpc)“ when executing Build.
  Use these files to convert a project.

  1. Select menu [Project] > [Open Project...] in CS+ and specify the “Renesas common project files (*.rcpc)” output by e^2 studio.
  2. The imported projects can be used in CS+.

Project Conversion Combinations

Converting to the Same Toolchain

- RX Family

  RX family C/C++ V1.00.00 or later versions are supported.

Output of Renesas Common Project	Reading of Renesas Common Project
CubeSuite+ V2.00.00～V2.02.01 CS+ V3.00 and later versions	e² studio V.3.0.0 and later versions
e² studio V.3.0.0 and later versions	CubeSuite+ V2.00.00～V2.02.01 CS+ V3.00 and later versions

- RL78 Family

  RL78 family C Compiler CC-RL V1.00.00 or later versions are supported.

Output Renesas Common Project	Reading of Renesas Common Project
CS+ V3.00 and later versions	e² studio V.4.0.0 and later versions
e² studio V.4.1.0 and later versions	CubeSuite+ V2.00.00～V2.02.01 CS+ V3.00 and later versions

- Other Families

  Not supported.

Project Conversion Combinations

Converting to a Different Toolchain

- RX Family
  Conversion of a project created through a toolchain other than the RX family C/C++ compiler to a project using the RX family C/C++ compiler.
  - Not supported.
- RL78 Family
  Conversion of a project created through RL78 family C compiler CA78K0R to a project using the RL78 family C compiler CC-RL.
  - e^2 studio does not support RL78 family C compiler CA78K0R.

Output Renesas Common Project	Reading of Renesas Common Project
CS+ V3.00 and later versions	e^2 studio V.5.1.0 and later versions

* This project conversion uses CS+ project file (*.mtpj).

* Note that source files are not converted. For source file conversion, refer to the following website.
Assistance in Porting Files to the C Compiler Package for RL78 Family | Renesas

- Other Families
  Conversion to a project using a different toolchain
  - Not supported

Restrictions

- This section shows the code or settings that cause differences in operation from that in the original build environment as a result of project conversion, and describes a workaround for the problem that may occur due to each difference.
- The descriptions also include preventive actions that should be taken before conversion; please be sure to read through this section before project conversion.
- Under “Reason” on each page, the reason for the corresponding restriction and tips for avoiding problems are provided; please also read this information.

Restriction on Output File of Compiler / Assembler

Restriction	e² studio -> CS+	CS+ -> e² studio
A build error may occur when importing a project which has output folder names or output file names of object file for Compiler / Assembler are changed.	Applied	Applied

Applied/Not applied:
- Applied: This restriction is applied to the conversion in the direction of the arrow.
- Not applied: This restriction is not applied to the conversion in the direction of the arrow.

- Workaround
  - Before importing a project, revert the output folders and output file names of object file for Compiler / Assembler to default.
- Reason
  - Each IDE converts the folder name and file name of the object file output by Compiler / Assembler according to the IDE specifications.
    - CS+ converts the output folder configuration of all source files to a single folder output.
    - e² studio converts the output folder configuration of all source files to folders in the same hierarchical structure of the source files.
  - The above restriction applies due to the difference in specifications between the tools.

Restriction on Output File of Linker

Restriction	e² studio -> CS+	CS+ -> e² studio
A build error may occur when importing a project which has output folder name or output file name of Linker is changed.	Applied	Applied

- Workaround
  - Before importing a project, revert the output folder and output file name of Linker to default.
- Reason
  - Each IDE converts the folder name and file name of the file output by Linker according to the IDE specifications.
    - e² studio manages output folder and output file name of Linker as Build Artifact.
    - CS+ manages output folder name and output file name of Linker as options of Linker.
  - The above restriction applies due to the difference in specifications between the tools.

Restriction on Folder Structure

Restriction	e² studio -> CS+	CS+ -> e² studio
The folder structure is ported the structure same as the previous IDE configuration although the handling of folders changes according to the IDE specifications.	Applied	Applied

- Workaround
  - After importing a project, reconfigure the folder structure if necessary.
- Reason
  - The display method of the folder structure differs depending on the specifications between the tools although the folder structure is kept. (Refer to the next page for example of the display method of the folder structure)
    - CS+ ports physical folder / virtual folder / linked folder of e² studio to the category items as virtual folder. And CS+ does not display the folders and files which are not used by Build.
    - e² studio ports the category items to physical folder when there is physical folder same location as category item. If there is not physical folder same location as category item, e² studio ports the category items to virtual folder. e² studio shows all physical folders and files under the project folder to project tree. Files which is not used by Build are shown the file as "Exclude from build".

Restriction on Folder Structure (Example 1)

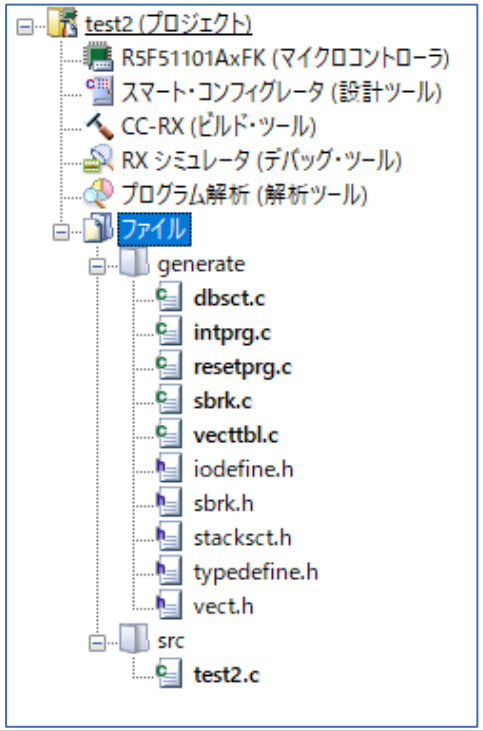
1. Actual folder / file structure

- ¥<Project folder>

  ¥dbsect.c ¥intprg.c ¥restprg.c
  ¥sbrk.c ¥vecttbl.c ¥iodefine.h
  ¥sbrk.h ¥stacksct.h ¥typedefine.h
  ¥vect.h ¥test2.c
  ¥hwsetup.c (Exclude from project)
  ¥lowlvl.src (Exclude from project)
  ¥lowsrc.c (Exclude from project)
  ¥lowsrc.h (Exclude from project)

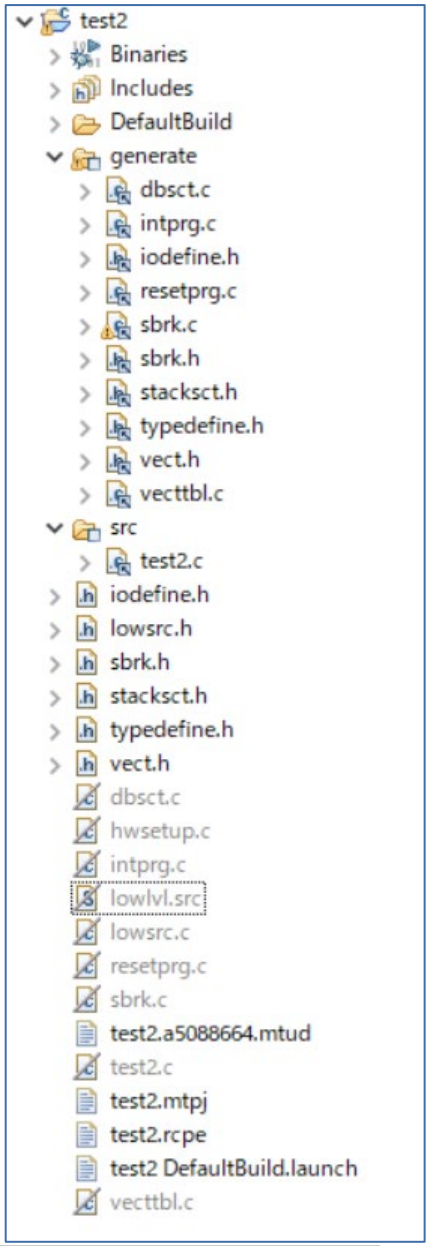
2. Create a project on CS+

Create “generate” category and “src” category as virtual folder. And register the source files on project folder to “generate” category and “src” category.



3. Import a project with e² studio

- Port “Generate” and “src” category to virtual folder.
- Port source files in each category to linked file.
- Port actual files in project folder to actual file as exclude from Build.



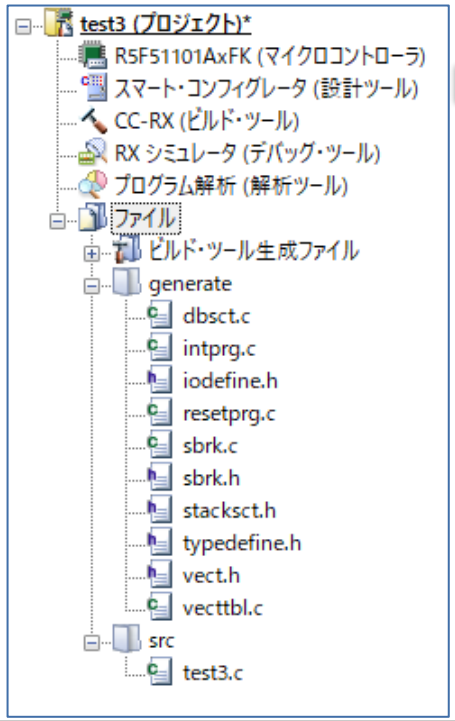
Restriction on Folder Structure (Example 2)

1. Actual folder / file structure

- ¥<Project folder>
  - ¥startup (folder)
    - ¥dbsect.c ¥intprg.c ¥restprg.c
    - ¥sbrk.c ¥vecttbl.c ¥iodefine.h
    - ¥sbrk.h ¥stacksct.h ¥typedefine.h
    - ¥vect.h
  - ¥src (folder)
    - ¥test3.c
    - ¥hwsetup.c (Exclude from project)
    - ¥lowlvl.src (Exclude from project)
    - ¥lowsrc.c (Exclude from project)
    - ¥lowsrc.h (Exclude from project)

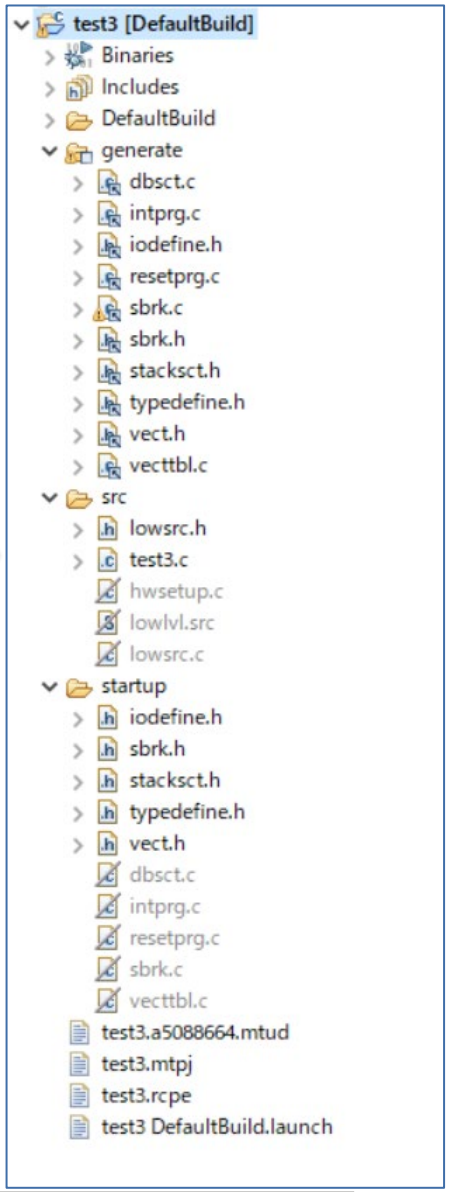
2. Create a project on CS+

Create “generate” category and “src” category as virtual folder. Then register source files on startup folder as actual folder to “generate” category and register source files on src folder as actual folder to “src” category.



3. Import a project with e² studio

- Port “generate” category to virtual folder.
- Port source files in “generate” category to linked file.
- Port “src” category to actual folder.
- Port source files in “src” category to actual file.
- Port startup folder to actual folder but source files are excluded from Build.



Restriction on Same File Name

Restriction	e² studio -> CS+	CS+ -> e² studio
A build error may occur when a project has duplicate file names although project conversion can be properly.	Applied	Not applied

- Workaround
  - Before importing a project, rename the file so that there are no duplicate file names.
- Reason
  - In CS+, duplicate file name cannot be specified. CS+ outputs object files to a single folder so that object files are conflict when there are duplicate source file name.
  - In e² studio, duplicate file name can be specified. e² studio outputs object files to folders in the same hierarchical structure of the source files. So, object files are not conflict.
  - The above restriction applies due to the difference in specifications between the tools.

Restriction on Folder Name and File Name

Restriction	e² studio -> CS+	CS+ -> e² studio
When a file name or a folder name uses a prohibited character (#, $, %), the project may not be ported correctly.	Applied	Applied

- Workaround
  - Before importing a project, change a file name or a folder name to not use a prohibited character (#, $, %).
- Reason
  - The characters (#, $) indicating build variables in e² studio and those (%) indicating placeholders in CS+ have special meaning and must not be used in folder names or file names.

Restriction on Linkage Object File / Library File

Restriction	e² studio -> CS+	CS+ -> e² studio
The library files and object files registered in a project are not ported as the files to be linked.	Not applied	Applied

- Workaround
  - Before importing a project, specify library files and object files in the option setting panel for Linker.
- Reason
  - e² studio does not automatically link the library files and object files registered in a project. They should be explicitly specified in the option setting panel.
  - CS+ automatically links the library files and object files registered in a project. In addition, it also links the files specified in the option setting panel.
  - The above restriction applies due to the difference in specifications between the tools.

Restriction on Additional Command for before/after Each Phase

Restriction	e² studio -> CS+	CS+ -> e² studio
Added commands to be executed before or after each processing phase (such as the compiler phase or assembler phase) in the build process cannot be ported to the target tool.	Not applied	Applied

- Workaround
  - After importing a project, add your commands according to each IDE specifications.
- Reason
  - e² studio allows commands to be added before or after the build process but does not allow commands before and after each phase in the build process.
  - CS+ allows commands to be added before or after the build process, and also before or after each phase in the build process.
  - The above restriction applies due to the difference in specifications between the tools.

Restriction on Additional Command for before/after Build

Restriction	e² studio -> CS+	CS+ -> e² studio
Commands written in multiple lines cannot be ported to the target tool although additional command for before/after Build is ported properly.	Not applied	Applied

- Workaround
  - Before importing a project, when commands are written in multiple lines, write them in a single BAT file and add the file to the build process.
- Reason
  - In e² studio, you need to specify '&' as separate character when adding multiple command lines.
  - CS+ allows commands in multiple lines.
  - The above restriction applies due to the difference in specifications between the tools.

Restriction on Additional Python Console Command for Build

Restriction	e² studio -> CS+	CS+ -> e² studio
Commands that can be executed on the Python console cannot be ported to the target tool.	Not applied	Applied

- Workaround
  - None
- Reason
  - e² studio does not support the build command via Python commands.
  - In CS+, commands that can be executed on the Python console can be added as commands to be executed during process.
  - The above restriction applies due to the difference in specifications between the tools.

Restriction on Debugging Configuration

Restriction	e² studio -> CS+	CS+ -> e² studio
Debugging configuration cannot be ported to the target tool.	Applied	Applied

- Workaround
  - After importing a project, add debugging configuration.
- Reason
  - Debugging configuration are not included in the target of project conversion.

Conversion of Build Variables and Placeholders

Conversion Table of Placeholders and Build Variables (1/2)

CS+ projects into e² studio

CS+ Placeholder	Meaning	e² studio Build Variable
%FilePath%	Absolute path to the file	Convert to “${selected_resource_loc}”. Please rewrite it to the necessary value for you.
%FileRelativePath%	Relative path from the project to the file	Convert to “${selected_resource_loc}”. Please rewrite it to the necessary value for you.
%FileDir%	Absolute path to the folder including the file	Convert to “${selected_resource_loc}”. Please rewrite it to the necessary value for you.
%FileName%	File name (not including the path)	Convert to “${selected_resource_name}”. Please rewrite it to the necessary value for you.。
%FileLeaf%	File name without file extension (not including the path)	Convert to “$(basename $(<F))”.
%FileExt%	File extension	Convert to “$(suffix $(<F))”.
%MainProjectDir%	Absolute path to the folder for the main project	Convert to “${ProjDirPath}”.
%MainProjectName%	Main project name	Convert to “${ProjName}”.
%ProjectDir%	Absolute path to the folder for the project	Convert to “${ProjDirPath}”.
%ProjectName%	Project name	Convert to “${ProjName}”.
%BuildModeName%	Build mode name	Convert to “${ConfigName}”.
%ConfigDir%	Absolute path to the folder for the configuration	Convert to “${workspace_loc:/${ProjName}/${ConfigName}}”.

Conversion Table of Placeholders and Build Variables (2/2)

CS+ projects into e² studio

CS+ Placeholder	Meaning	e² studio Build Variable
%MicomToolPath%	Absolute path to the folder where CS+ is installed	Convert to “${MicomToolPath}”. e² studio does not have the compatible Build Variable with that Placeholder. Please rewrite it to the necessary value for you.
%TempDir%	Absolute path to the temporary folder	Convert to “${TEMP}”.
%WinDir%	Absolute path to the Windows system folder	Convert to “${windir}”.
%ActiveProjectDir%	Absolute path to the folder for the active project	Convert to “${ProjDirPath}”. e² studio does not have the compatible Build Variable with that Placeholder. Please rewrite it to the necessary value for you.
%ActiveProjectName%	Active project name	Convert to “${ProjName}”. e² studio does not have the compatible Build Variable with that Placeholder. Please rewrite it to the necessary value for you.
%MainProjectMicomName%	Name of the target microcontroller for the main project	Convert to “${MainProjectMicomName}”. e² studio does not have the compatible Build Variable with that Placeholder. Please rewrite it to the necessary value for you.
%ProjectMicomName%	Name of the target microcontroller for the project	Convert to “${ProjectMicomName}”. e² studio does not have the compatible Build Variable with that Placeholder. Please rewrite it to the necessary value for you.
%ActiveProjectMicomName%	Name of the target microcontroller for the active project	Convert to “${ActiveProjectMicomName}”. e² studio does not have the compatible Build Variable with that Placeholder. Please rewrite it to the necessary value for you.

- If the option to specify a directory as a parameter has a Placeholder that indicates something other than a directory, convert to "${ProjDirPath}/<Build Variable>".

Conversion Table of Build Variables and Placeholders (1/2)

e² studio projects into CS+

e² studio Build Variable	Meaning	CS+ Placeholder
${workspace_loc}	Absolute path to the workspace folder	Convert to “%ProjectFolder%/../”.
${workspace_loc:/<path>}	Absolute path to the workspace folder ¥<path>	Convert to “%ProjectFolder%¥..¥<path>”.
${workspace_loc:/${ProjName}/<path>}	Absolute path to the project folder ¥<path>	Convert to “%ProjectFolder%¥..¥%ProjectName%¥<path>”.
${WorkspaceDirPath}	Absolute path to the workspace folder	Convert to “%ProjectFolder%/..”.
${ProjDirPath}	Absolute path to the project folder	Convert to “%ProjectFolder%”.
${ProjName}	Project name	Convert to “%ProjectName%”.
${CONFIGDIR}	Absolute path to the configuration folder	Convert to “%CondifDir%”.
${ConfigName}	Configuration name	Convert to “%BuildModeName%”.
${selected_resource_loc}	Absolute path to the selected resource folder	Convert to “%FullFile%”. CS+ does not have the compatible Placeholder with that Build Variable. Please rewrite it to the necessary value for you.
${selected_resource_name}	Selected resource name	Convert to “%FileName%”. CS+ does not have the compatible Placeholder with that Build Variable. Please rewrite it to the necessary value for you.
${selected_resource_path}	Relative path from the workspace folder to the selected resource folder	Convert to “${selected_resource_path}”. CS+ does not have the compatible Placeholder with that Build Variable. Please rewrite it to the necessary value for you.

Conversion Table of Build Variables and Placeholders (2/2)

e² studio projects into CS+

e² studio Build Variable	Meaning	CS+ Placeholder
${eclipse_home}	Absolute path to the folder where e² studio is installed	Convert to “${eclipse_home}”. CS+ does not have the compatible Placeholder with that Build Variable. Please rewrite it to the necessary value for you.
${TEMP}	Absolute path to the temporary folder	Convert to “%TempDir%”.
${TMP}	Absolute path to the temporary folder	Remove “${TMP}”. CS+ does not have the compatible Placeholder with that Build Variable. Please add the necessary value for you.
${windir}	Absolute path to the Windows system folder	Convert to “%WinDir%”.

Renesas.com