

Kinetis Design Studio Porting Guide

1 Introduction

The CodeWarrior for Kinetis Design Studio Porting migration assistant helps in migrating legacy Kinetis CodeWarrior GNU/gcc projects to projects configurable in Kinetis Design Studio. To successfully build an imported projects, you need to manually make some changes to the project content and files. This document lists the steps to convert or import an existing Kinetis project to a Kinetis Design Studio project.

2 Create a new project or import an existing project

The process to import and convert projects is a semi-automated process. This is because the projects settings between CodeWarrior and Kinetis Design Studio are different, the import process outlined in the following sections tries to map the CodeWarrior settings to the Kinetis Design Studio settings.

However, as every project is different and can have different settings, the importer/converter might not be able to properly convert every project, especially if your CodeWarrior projects is highly customized.

In case you trying to import\convert a highly customized project, perform the following steps:

© 2015 Freescale Semiconductor, Inc.

Contents

1	Introduction.....	1
2	Create a new project or import an existing project.....	1
3	Importing an existing project.....	2
4	Converting a legacy Kinetis project.....	5
5	Identifying the conversion errors and warnings.....	7
6	Manually updating and cleaning the converted project.....	10
7	Building the converted project.....	11

Importing an existing project

1. Create a new project, select File > New > Kinetis Project in Kinetis Design Studio.
2. Add your own sources to the new project.
3. Apply tool chain/build options, if appropriate.

3 Importing an existing project

To import an existing Kinetis project:

NOTE

CodeWarrior for Microcontrollers offers two compilers: legacy Freescale compiler, for Kinetis-K devices and GNU gcc for Kinetis-K and Kinetis-L devices. The legacy Freescale compiler projects settings are very different from the projects settings in Kinetis Design Studio. Therefore, Kinetis Design Studio can import/convert only the CodeWarrior GNU/gcc compiler projects.

1. Select **File > Import**, from the IDE menu.

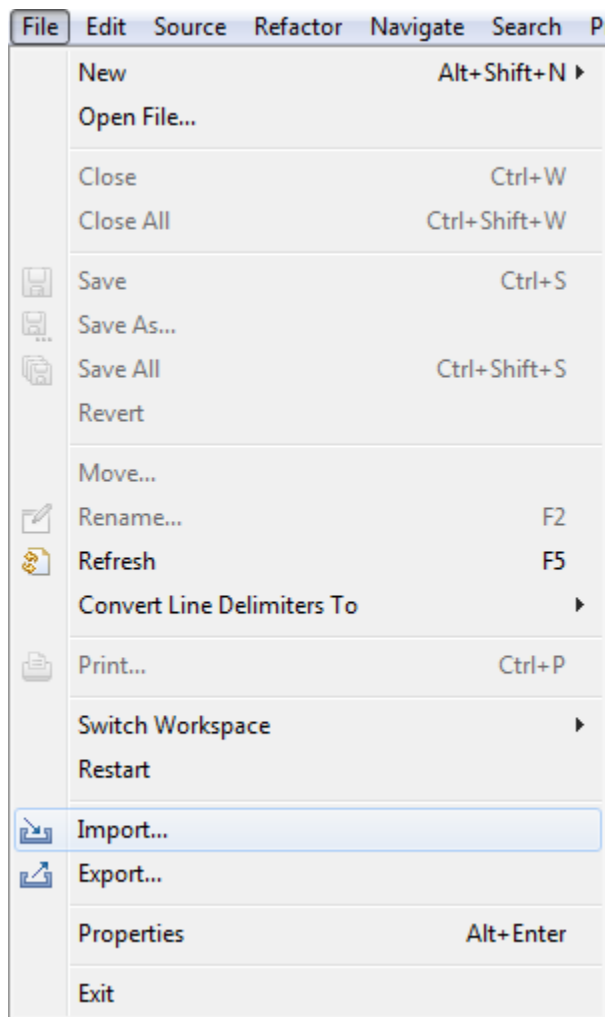


Figure 1. Select import option

The import dialog appears.

2. Expand the **General** tree and select **Existing Projects into Workspace**.

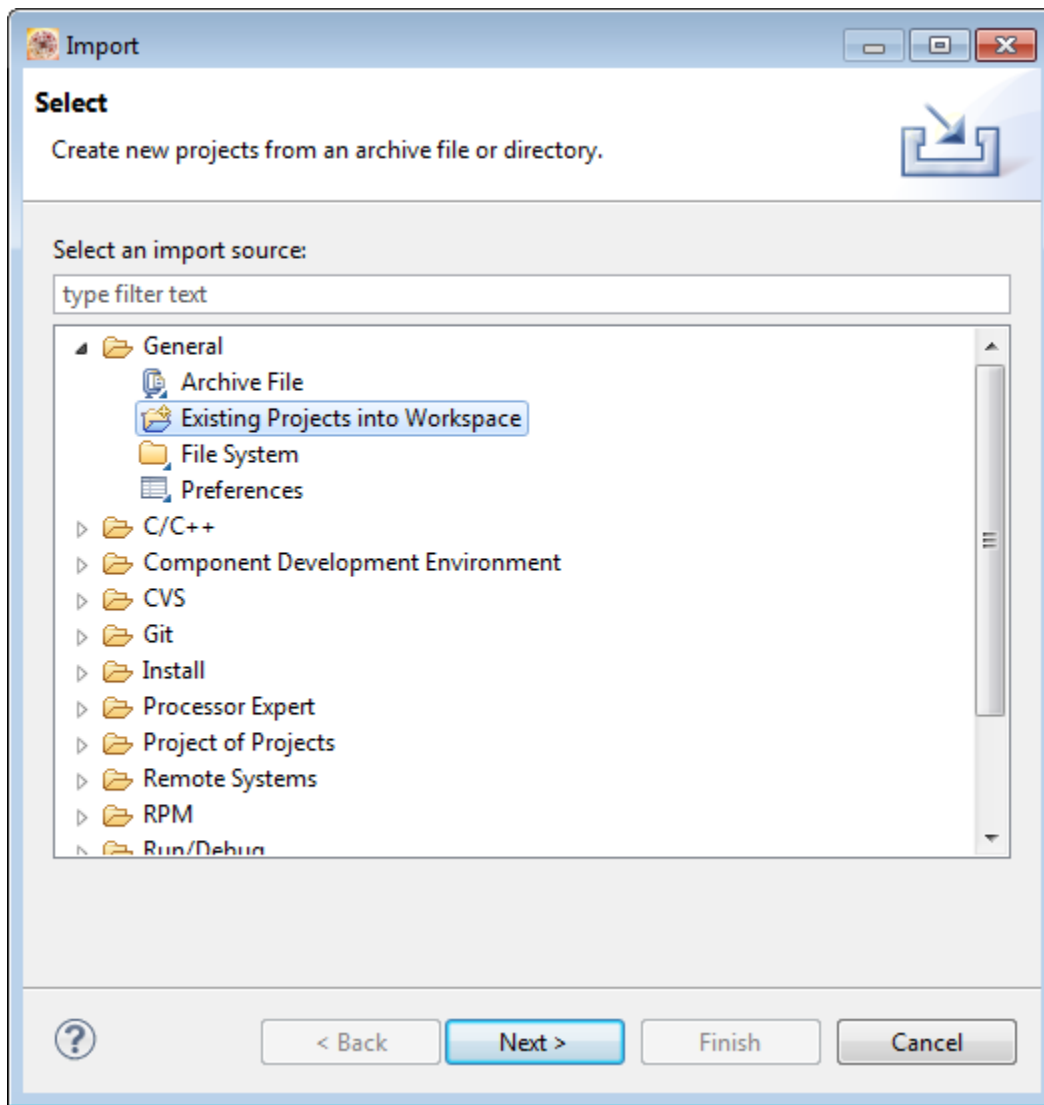


Figure 2. Select an import source

3. Click **Next**.
4. Click **Browse** and select the root directory to search for an existing Eclipse project. You can also choose to import an archive file.
5. Check the **Copy projects into workspace** checkbox. This will import your existing project in the current Workspace. Changes made to the project in your Workspace will not impact your existing project or workspace makes a physical copy of your project. If your project contains links to other folders or locations, you will need to change them later accordingly.

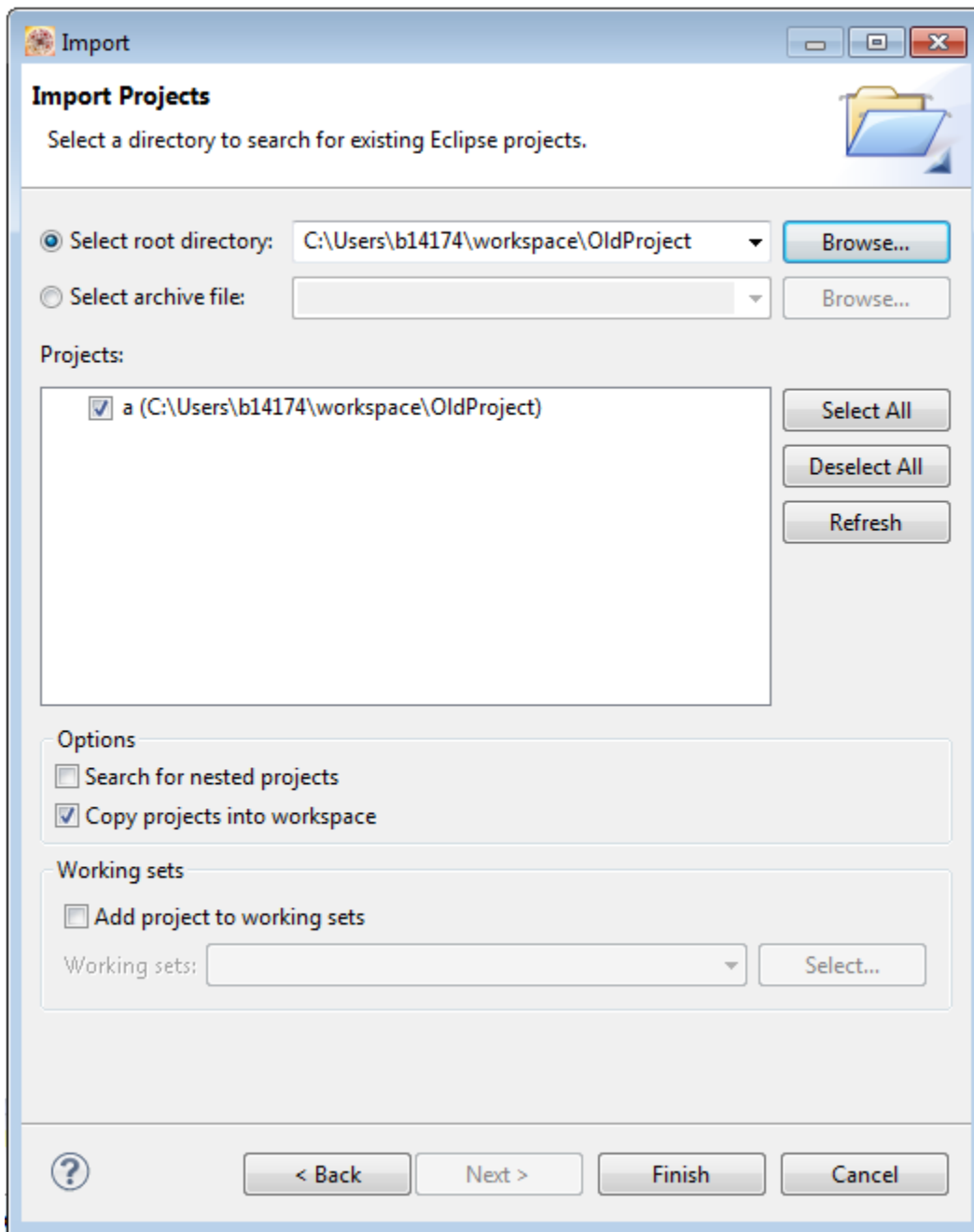


Figure 3. Copy project in workspace

6. Click Finish.
The imported project appears in the **Project Explorer** view.

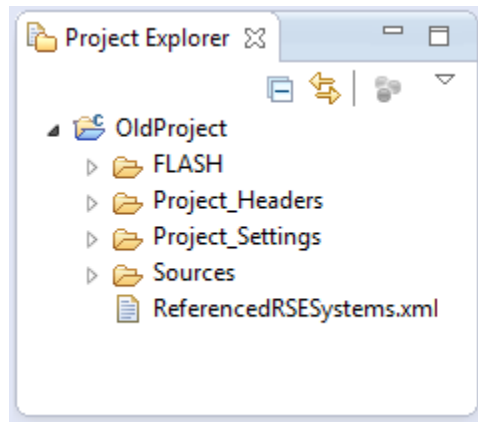


Figure 4. Project Explorer view

4 Converting a legacy Kinetis project

To use the imported legacy Kinetis project in Kinetis Design Studio, you need to convert it. The steps to convert a legacy Kinetis project are:

1. Select the recently imported legacy project you want to convert.

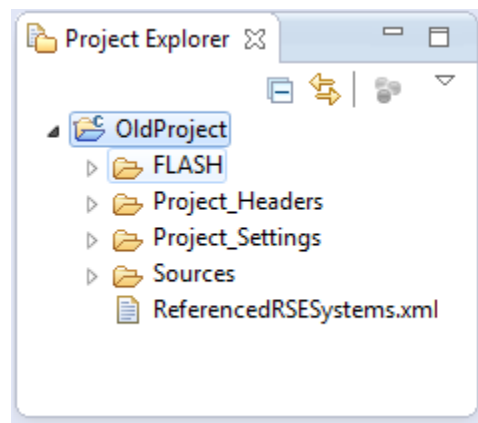


Figure 5. Select the project

2. Right-click and select **Convert CodeWarrior Project file**.

NOTE

The context menu **Convert CodeWarrior project file...** will only work with GNU/gcc CodeWarrior project and not for CodeWarrior projects using the legacy Freescale compiler.

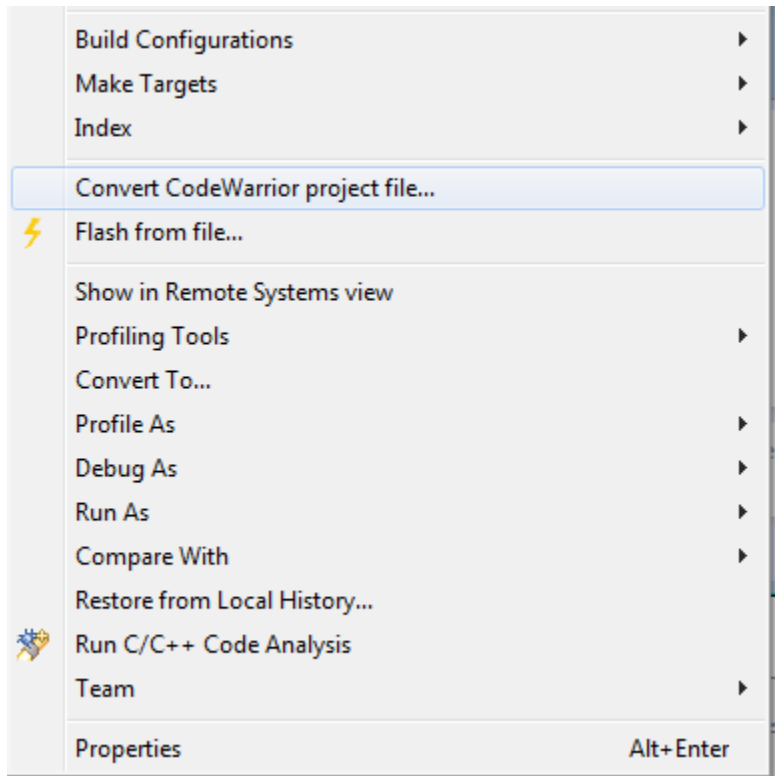


Figure 6. Select Convert CodeWarrior Project file

The CodeWarrior to KDS project migration assistant dialog appears and prompts whether you want to convert the project.

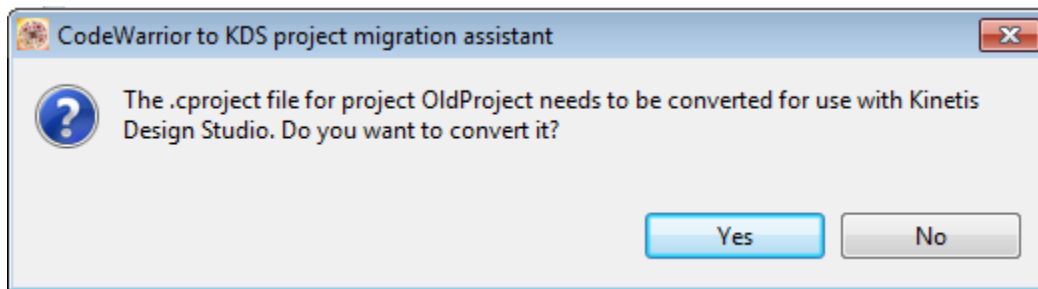


Figure 7. CodeWarrior to KDS project migration assistant dialog

3. Click Yes.

The CodeWarrior to KDS project migration assistant will prompt that the conversion has completed and a backup of the original *cproject file has been created in .cproject_backup.

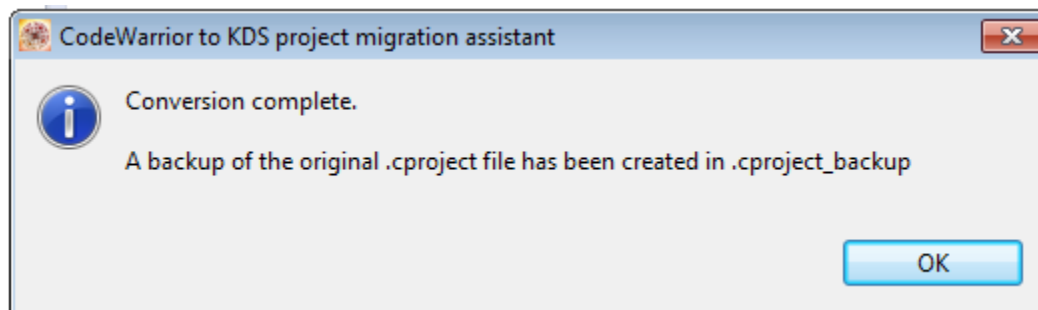


Figure 8. Conversion completed

4. Click **OK** to close the CodeWarrior to KDS project migration assistant dialog.

NOTE

>To see the conversion process log, navigate to the project's root directory and open the KDSConverter.log file.

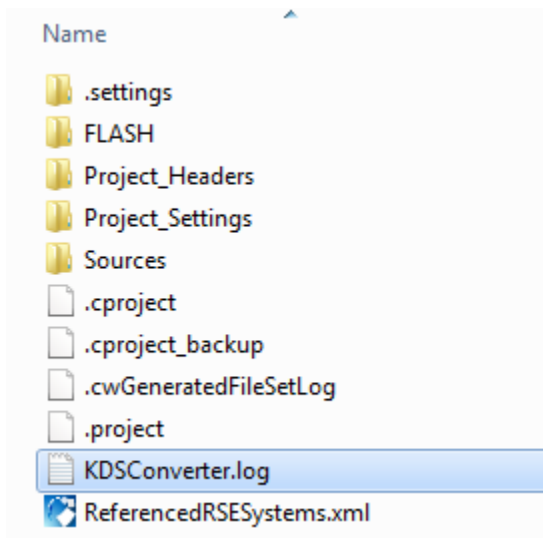


Figure 9. Conversion log

5 Identifying the conversion errors and warnings

To ensure that you can successfully build and configure the converted project, you need to first identify the errors and warnings encountered during the conversion.

The errors and warnings encountered during conversion are listed in the **Problems** view.

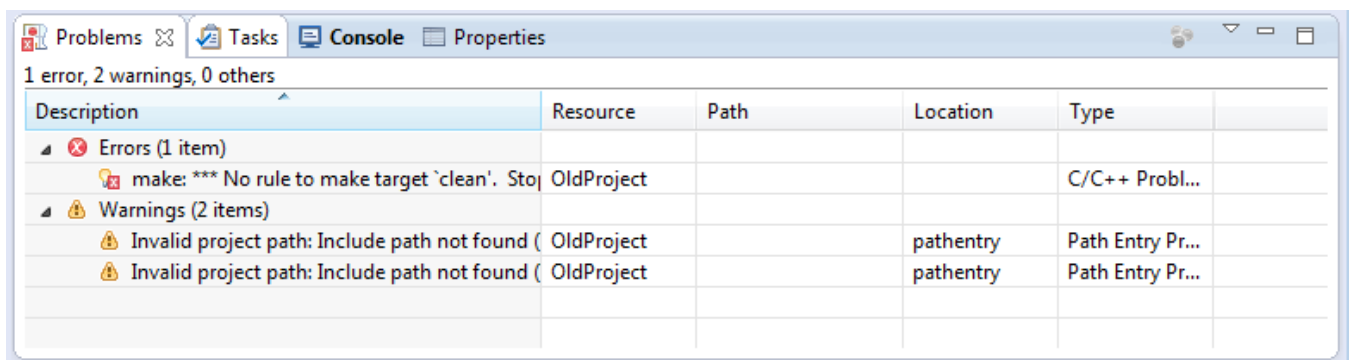


Figure 10. Problems view

NOTE

The Problems view appears in the IDE by default. However, you can also access and open the **Problems** view by selecting **Window > Show View > Problems** from the IDE menu bar.

Identifying the conversion errors and warnings

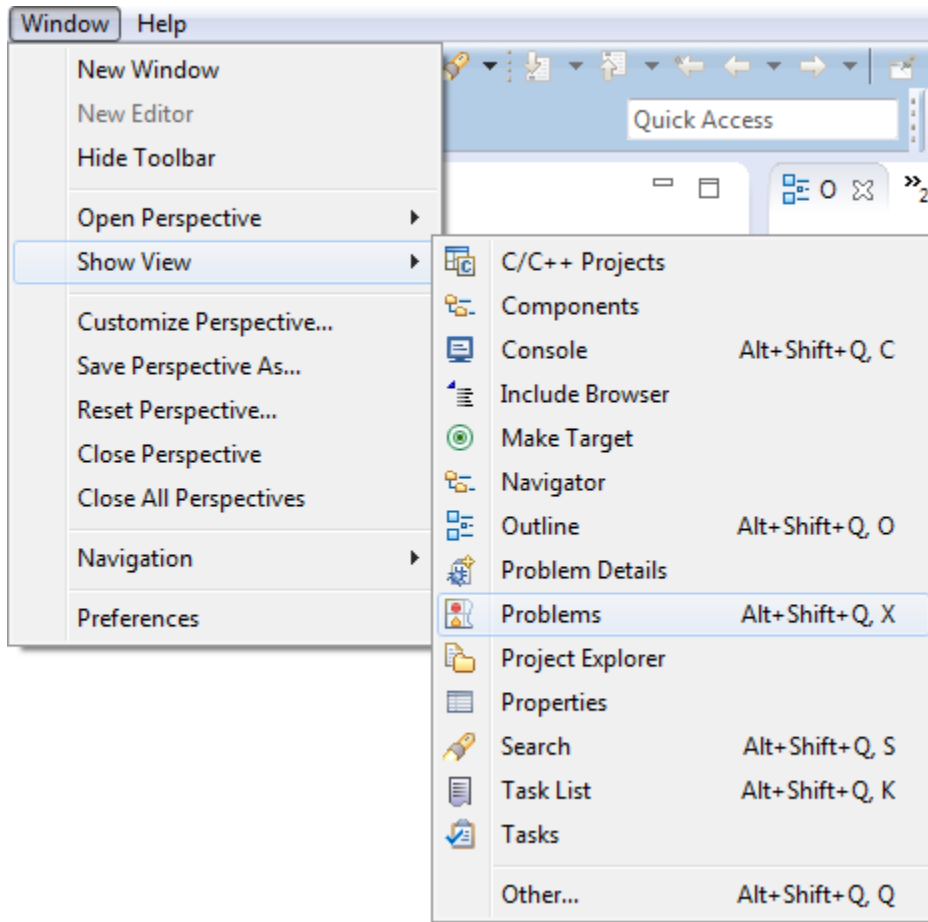


Figure 11. Accessing the Problems view

The complete list of errors and warnings encountered in the Workspace is available in the **Error log** view.

To open the **Error Log** view:

1. Select **Windows > Show View > Other > Error Log**.

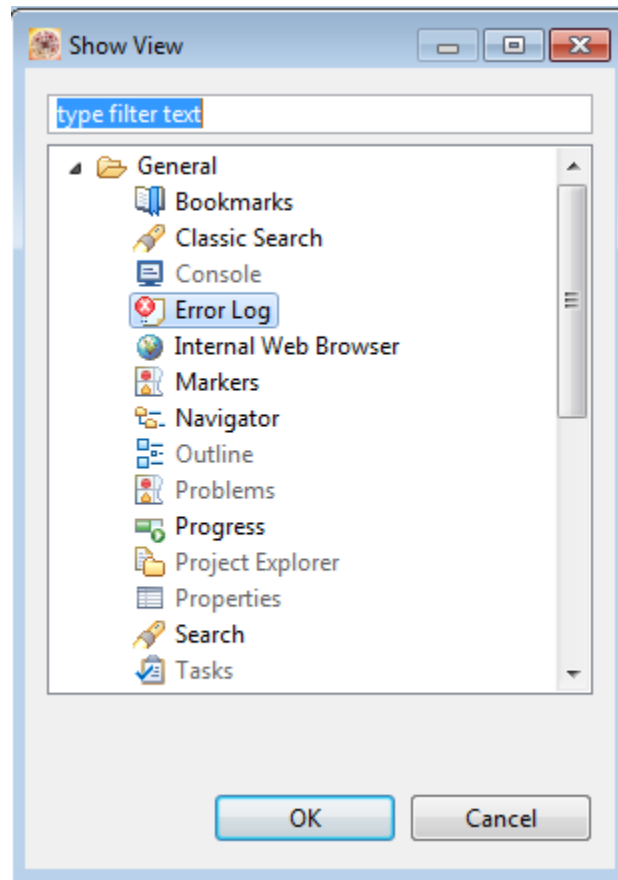


Figure 12. Accessing the Error log view

2. Click **OK**. The **Error Log** view appears stacked in the IDE.

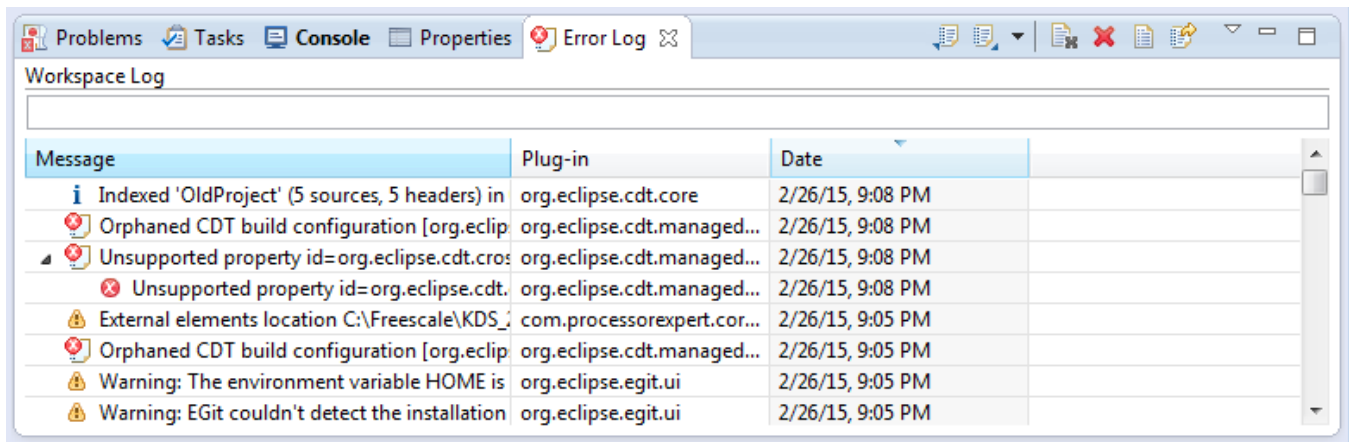


Figure 13. Error Log view

NOTE

These errors occurs if manual updates to the project are not done. However, once the update is done these errors will not go away (because they are stored in the log file) but should stop occurring each time the project is opened after the project has been updated.

6 Manually updating and cleaning the converted project

Once you have identified the issues

To clean a project:

1. Select the convert project from the Project Explorer view.
2. Specify the path for compiler include paths.
 - a. Right-click on the selected project and select **Properties**
 - b. Expand the **C/C++ Build** tree and select **Settings > Tool Settings**.
 - c. Navigate to **Cross ARM C Compiler** and select **Includes**.

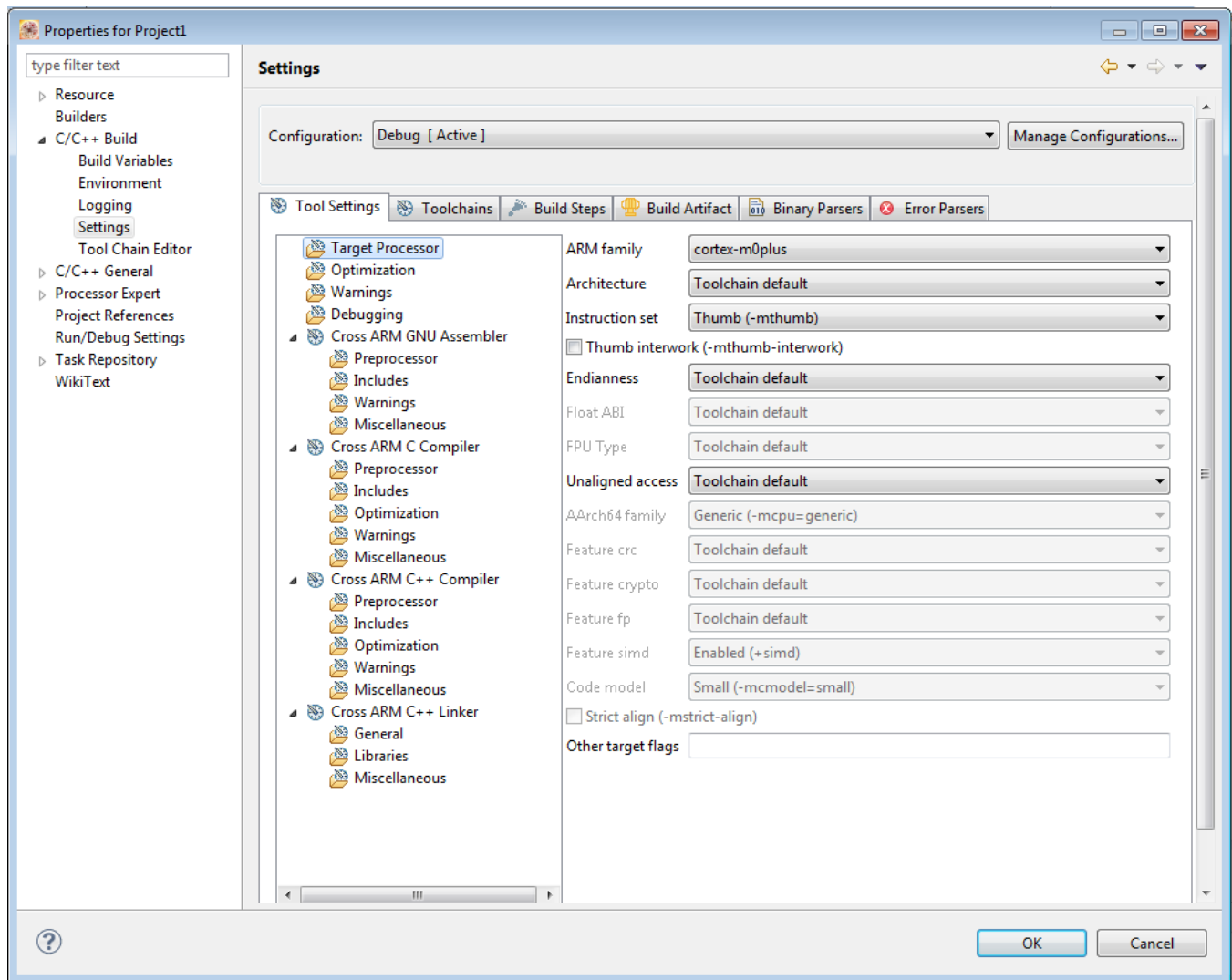



Figure 14. Build settings

- d. Click the  button and specify the absolute path of the libraries you want to include.

Alternatively, you can navigate to Project > Properties > C/C++ Build > Build Variables and define an environment variable `#{MCUToolsBaseDir}`. Defining the variable makes it accessible within the Eclipse environment. To define the variable, click **Add** and specify the variable name and Value. Check the **Apply to all configurations** checkbox if you want to access the variable from other configurations. Click **OK** to close the **Define a New Build Variable** dialog.

3. Navigate to Project > Properties > C/C++ Build > Settings > Cross ARM C Linker > Libraries > Libraries (-).
4. Add the library `rt`.
5. Click **OK** to close the **Properties** dialog.
6. Right-click on the project and select **Clean Project**.

7 Building the converted project

To build the project converted project:

1. Select the project in the Project Explorer view.
2. Select Project > Build Project.

Index

How to Reach Us:

Home Page:

freescale.com

Web Support:

freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. Freescale reserves the right to make changes without further notice to any products herein.

Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale, the Freescale logo, and Kinetis are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. ARM is the registered trademark of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All other product or service names are the property of their respective owners. All rights reserved.

© 2015, Freescale Semiconductor, Inc.