**Cyclone Project Template**

Table of Contents

[Chapter 1 2](#_Toc338945858)

[Introduction to the CCSv5.2 cyclone (UCD3138) Project Template 2](#_Toc338945859)

[Chapter 2 8](#_Toc338945864)

[Using Template for converting CCS3.3 Project to CCS5.2 project ….8](#_Toc338945865)

[Appendix 1 15](#_Toc338945872)

[Appendix 2 16](#_Toc338945873)

# Chapter 1

# Introduction to the CCSv5.2 cyclone (UCD3138) Project Template

Project Template in Code composer studio for Cyclone device is an empty project with predefined common project settings. The project template alone can't be build, because it does not have all the source files (like main.c, interrupt.c etc…). Template makes easier to for setting up a project for cyclone device in CCSv5.2. The template take care all necessary steps like creating target platform, common project settings, post processing setup etc…

This document explains the template structure and using the template for converting cyclone projects developed in CCSv3.3 to CCSv5.2.

You must have CCSv5.2 or later installed in your computer as prerequisite for completing this document. Also you must have CCSv3.3 project to be converted to CCSv5.2 with all file specific compile options if any.

**Attachment: Template\_UCD3138.**

The attached Template Project folder for Cyclone devices has the following files,

* Project Configuration files-- **.ccsproject**, **.cproject**, **.project**
* Target configuration files-- **.gel** and **.ccxml** file
* Device global variables definition file.(**cyclone\_global\_variables\_defs.c**)

These files, stores all project specific information, build commands and build options.

**Note: Please do not manually edit these files for normal usage with cyclone devices**

### Importing Cyclone Template into CCSv5.2

**Step 1: Open CCSv5.2**

Open CCSv5.2 from the installed directory (you can double click from desktop if an icon created during installation)

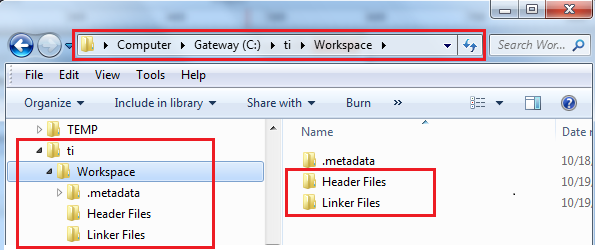
During first open CCSv5.2 will ask you to point a workspace (The ***Workspace*** is a directory which holds information about the projects you work on, as well as the Eclipse preferences (from layout to behavior)). Point your workspace in the local disk, it can be anywhere in the disk.

All projects which are not referenced will be saved in workspace location.

**Step 2: Setting up the Header files, Linker and Library files**

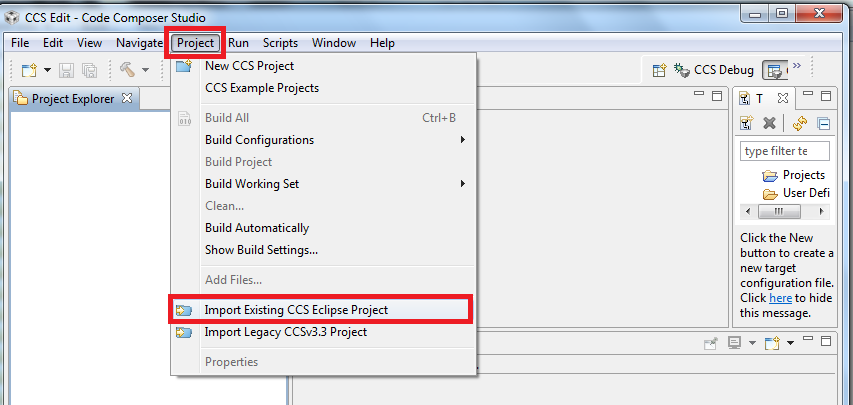
As per your development environment setup header files, library files and linker file location. In template we assume Header files and Linker files folders are in ***Workspace*** (Header file and linker file are in the folder where the project folder is located). Please refer Figure 1.

You can always change this based on your requirements, but we recommend keeping Header files, Linker in same location of project. In case if you chose different location for Header and linker files please update the search path in the template for this (will be explained later)

 Figure: 1

**Step 3: Importing cyclone CCSv5.2 project template into CCSv5.2 workspace.**

Go to Project menu and select Import Existing CCS Eclipse Project. Please Figure 2.

Figure: 2

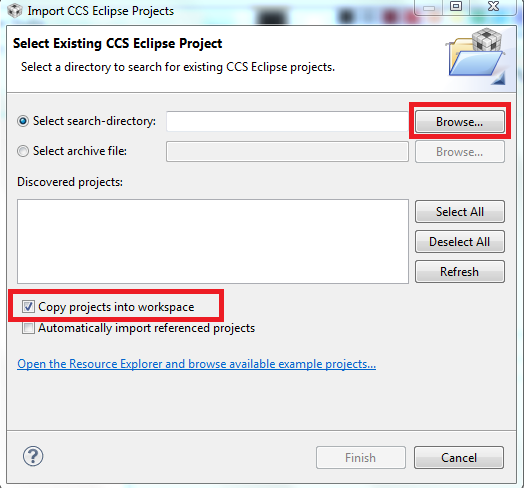
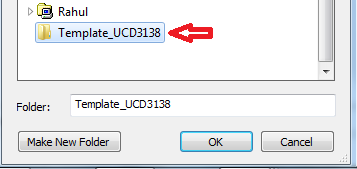
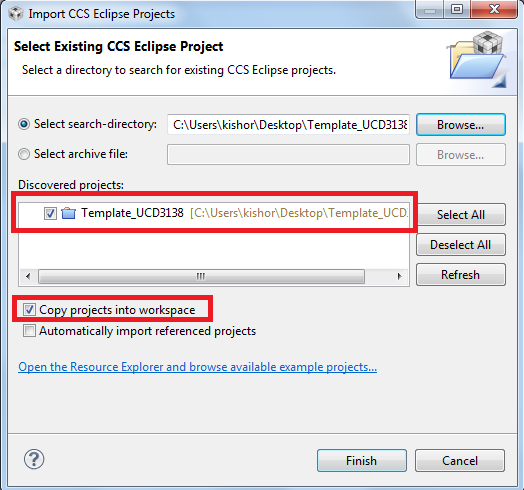
1. 

Figure: 3

1. Step 4: Select Copy project into workspace and click on Browse to and browse the project template. Refer figure: 3
2. **Step 5:** choose the project template from attached folder as seen the Figure 4
3. Figure: 4
4. 

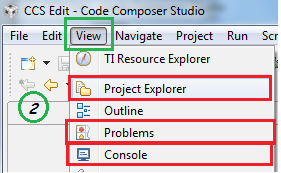
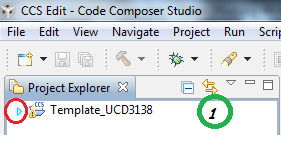
After clicking on *OK* you will get the bellow window. Make sure the Project is selected as shown in the Figure 5. In case if it is deselected, it means there is a folder in ***Workspace*** with same name (in this case Template\_UCD3138). Move/rename the folder from ***Workspace*** and try re-importing the Template from Step 3.

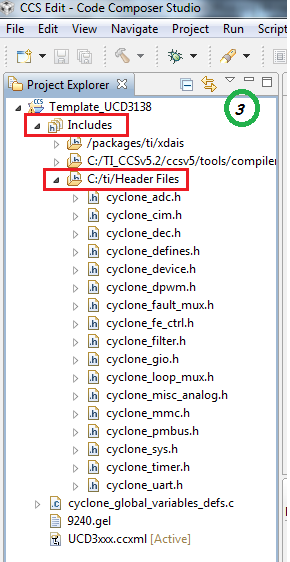
1. Figure: 5

Click on *finish*, then the CCS will start importing the Project (here Template\_UCD3138).

**Step 6:** After importing the Template you should see the project in CCS project explorer window as shown in bellow picture labeled 1 in Figure 5. If you can't see the project explorer window go to view menu and click project explorer (Refer bellow picture labeled 2 in Figure 5). You can expand the project folder from project explorer window as seen in Figure 6:

This finishes the importing template task, next chapter we explain how to import project files to template and build the project using CCSv5.2

Figure 5 

**Figure 6**

***Tip: Expand include folder make sure you can open any file double clicking on it, this confirm that there is no broken link to header file path.***

***Important notes:***  *1) Header Files Folder contains standard Device specific header files.*

***Device specific Header files*** *are,*

*cyclone\_adc.h*

*cyclone\_cim.h*

*cyclone\_dec.h*

*cyclone\_defines.h*

*cyclone\_device.h*

*cyclone\_dpwm.h*

*cyclone\_fault\_mux.h*

*cyclone\_fe\_ctrl.h*

*cyclone\_filter.h*

*cyclone\_gio.h*

*cyclone\_loop\_mux.h*

*cyclone\_misc\_analog.h*

*cyclone\_mmc.h*

*cyclone\_pmbus.h*

*cyclone\_sys.h*

*cyclone\_timer.h*

*cyclone\_uart.h*

Make sure that these files are there in Header *Files*Folder*.*

*2) Linker Files*Folder c*ontains Device specific linker command file and Library file*

***Device specific Linker command file****- cyclone\_headers.cmd*

**Library file**- rts32tiabi.lib

Confirm that these twofiles are in *Linker Files*Folder

**The current template is configured using RTS library rts32tiabi.lib. This library form TMS470 version (v4.1.4, from CCS3.3 ). There is a new arm7 TI library. For linking against new TI library for ARM please refer** [**Appendix 2.**](#_Appendix_2)

# Chapter 2

### Using Template for converting CCS3.3 Project to CCS5.2 project

Open the CCSv5.2, make sure that Cyclone template project is available in *Project* *Explorer* window.

You must have CCSv3.3 project folder to complete the exercise. For this demonstration we refereed Lab 01 project which is compiled in CCSv3.3.

*Note: For every Project conversion from CCSv3.3 to CCSv5.2, import the Template into the Workspace(Please refer Chapter 1 for Importing the Template).*

Step 1: Rename the project.

Right click on Project and select “*Rename*”, and then enter the name as you it wants to be. Please Figure 7

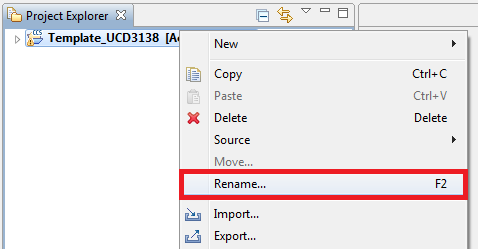
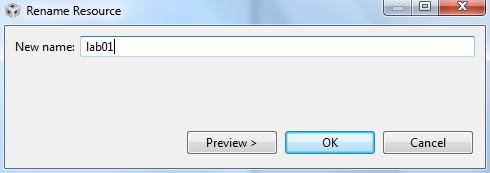
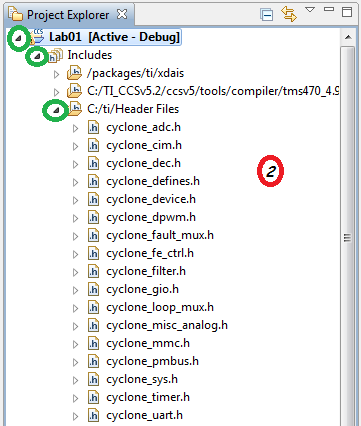
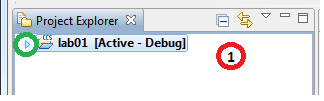


Figure 7

Note : If you add spaces in project name the post processing command will not generate x0 files with full project name. It will take only the first word from the project name and generate the x0 file with that word name. To avoid this either uses a project name without space in it or manually provide out file name in post process step. Please refer [appendix 1](#_Appendix_1) for details.

Now click on the expand button to expand the project folder to view project files.

Figure 8 Figure 9

Step 4: Adding files to the project.

Right click on the project and select Add *files*.

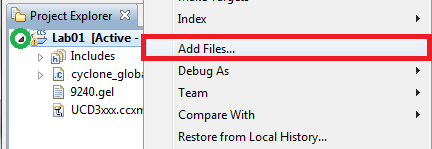
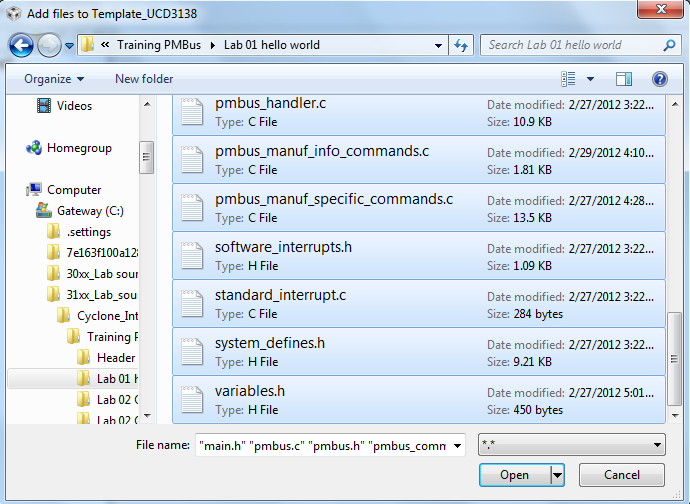


Figure 10

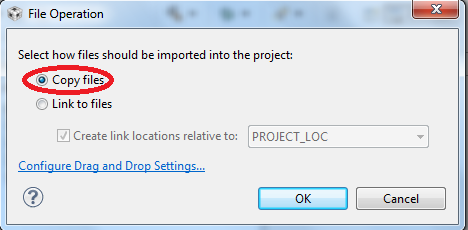
Now add *Project specific files* {Source files (**.c** or .**asm**), linker command files (**.cmd**) and header (**.h**) files}. Please refer bellow picture

**Note: Do not add *Device specific Header Files and Linker Files* to the project as they will be linking to the project. Library file is will also be linking to the project, so do not add it to the project. Pleases refer *Important Notes* section of Chapter 1.**

Figure 11

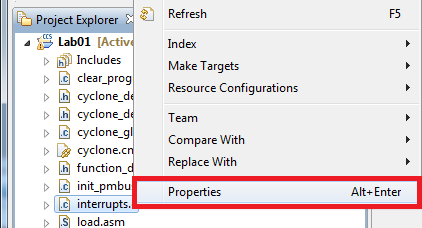
Then click open and in next window, select “*Copy* *files*” , click on *OK*.

Please refer Figure 12

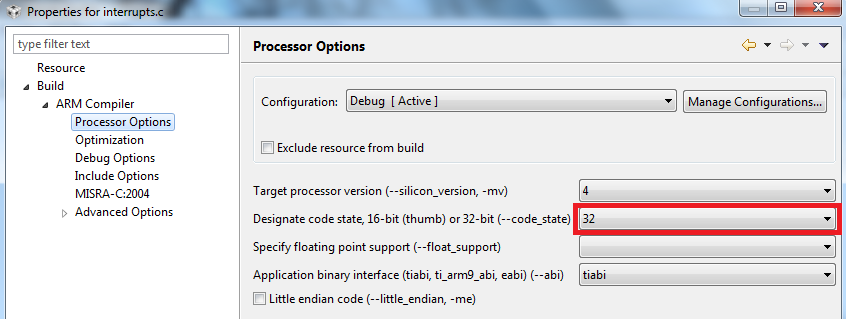
Figure 12

Step 5: File specific property settings.

Few files might needs to have specific properties other than common properties, say interrupt files needs to compile in ARM mode. For this right click on the file choose properties from context menu and add the desired compile options. Refer Figure 13 and 14 and 15 for choosing options for interrupt.

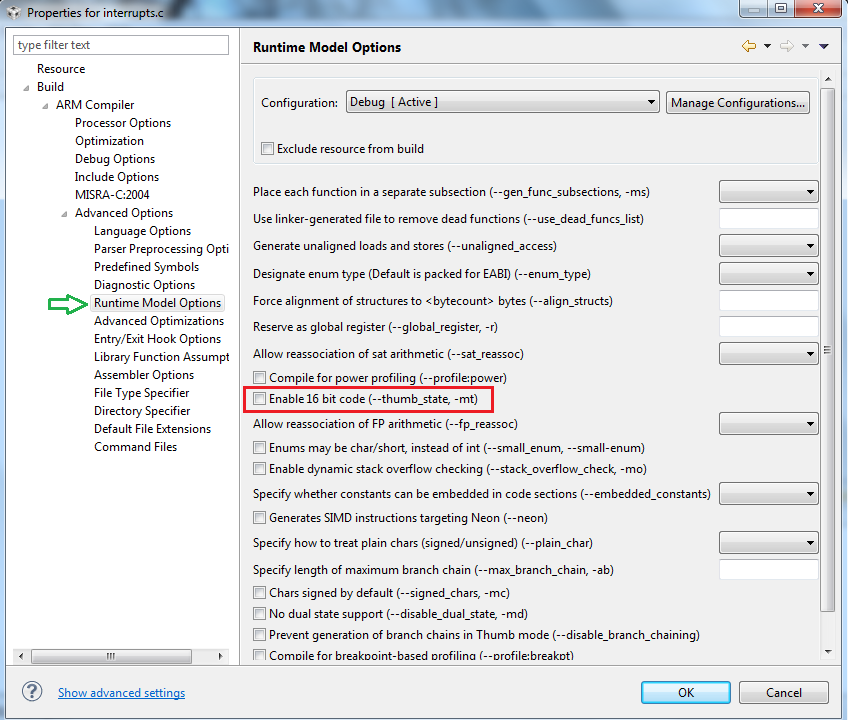
Figure 13

Go to *Processor Options* under ARM Compiler and select *designated code state* as **32**. This will tells the compiler to compile this specific file in 32 bit(ARM mode) mode. Please refer bellow picture



*Figure 14*

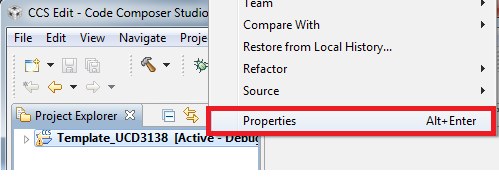
*Go to Advanced options under ARM Compiler, then deselect the Enable 16-bit code.* Refer Figure 15. Then Select *OK.*

Figure 15

Step 6: Linking the Header and linker files.

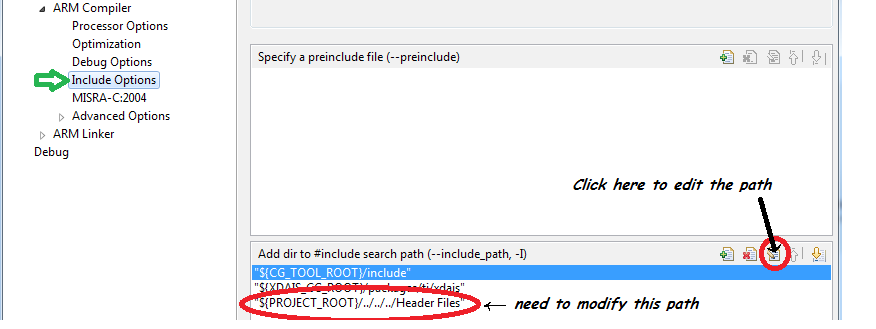
**If Header files and Linker file folders are linked to the workspace projects as specified in Chapter 1, you can skip this step**

Right click on the Project and select Properties. Refer Figure 16..

Figure 16

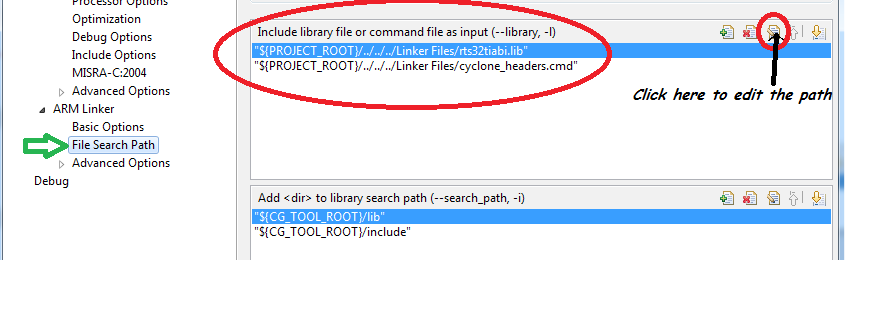
Add the *Include search path* in *Include Options* section. Refer the following picture.

*${PROJECT\_ROOT}/.../Header Files , It says that the Header Files folder is located at a directory back the Workspace. Refer the bellow picture and refer picture labeled as 1(bellow) for folder location.*



1. Figure 17
2. Add the *Include library file* of *command file* *r*oot in *File search path*. Figure 18 for reference.

${PROJECT\_ROOT}/../Linker Files/rts32tiabi.lib , It says that the Linker Files(It contains *rts32tiabi.lib* library file) folder is located at a directory back the Workspace. *Refer the bellow picture and refer bellow picture labeled as 1 for folder location .*

1. Figure : 18

Select *OK* after changing the path in Project properties.

1. *Step 7: Build* the project.

Right click on the Project and Select *Build Project. Please refer the following picture.*

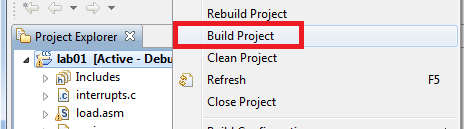


Figure 19

This should build the project and produce x0 file in the debug directory, you can see warnings and errors in case if any in the problems window.

# Appendix 1

To Add a custom name for post process command output , From the project explorer right click on the project folder , select properties from the context menu.

Under Steps tab, modify the command for Post-build steps. Only modify the output file name. here you can provide spaces if needed.

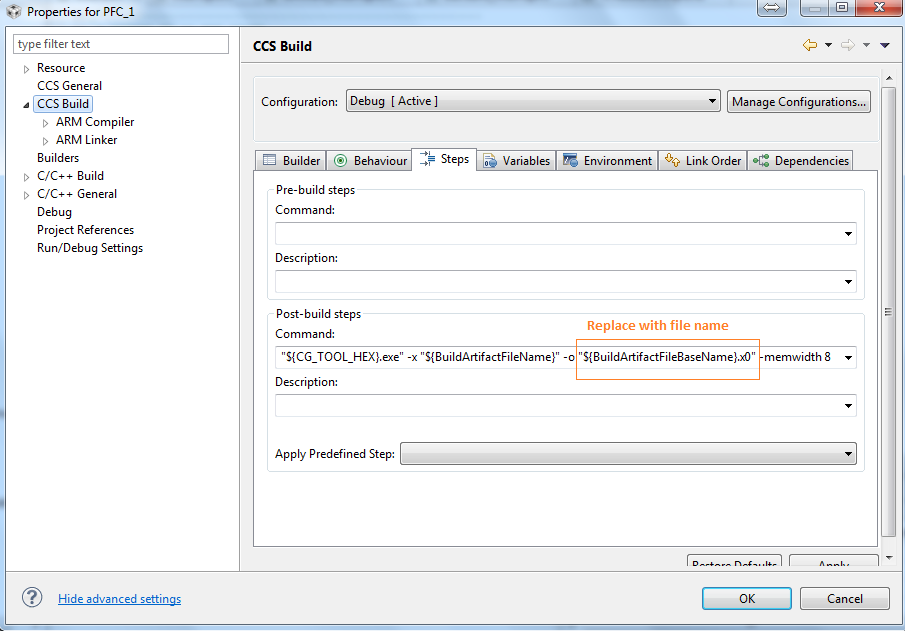


Figure 1

# Appendix 2

Linking against New TI ARM7 library:

**Step 1: Verifying new TI ARM7 library (*rtsv4\_T\_be\_tiarm7.lib*) is available in your installation directory.**

Go to Default path for Libraries **(<CCS5.2 INSTALL DIR>/ccs5/tools/compiler/tms470\_4.9.1/lib>)**

And check for the new TI ARM7 library's (***rtsv4\_T\_be\_tiarm7.lib***) availability.

If you do not see rtsv4\_T\_be\_tiarm7.lib in the tool path (refer Figure 1) then proceed with step 2 to install the libraries before you build. If you can see the library file rtsv4\_T\_be\_tiarm7.lib in the tool path (refer Figure 2) then you continue building the project with step 3

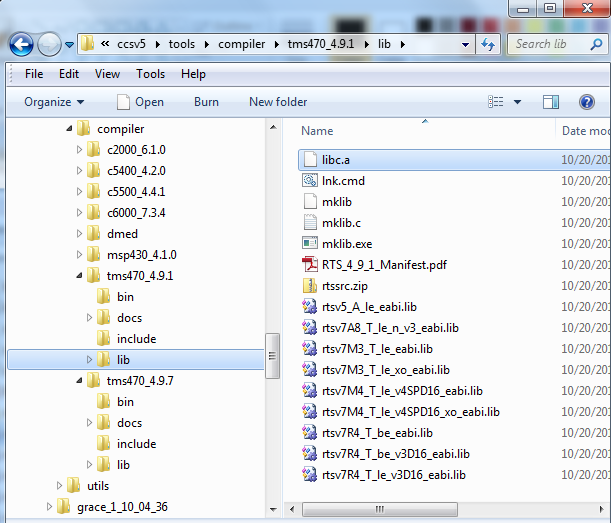


Figure 1

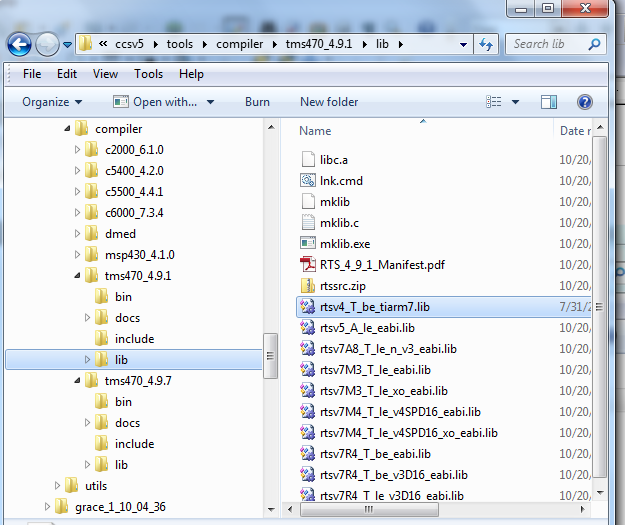
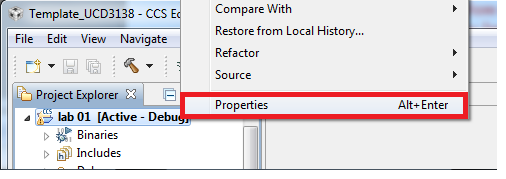


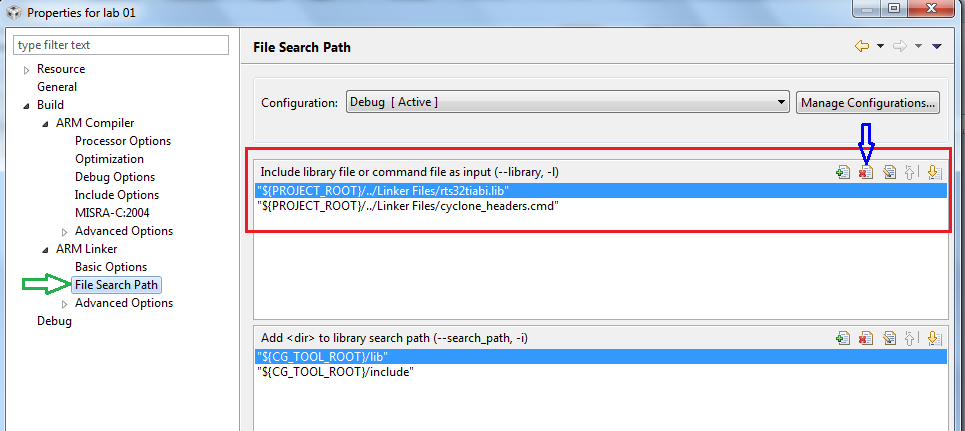
Figure 2

**Step 2 : Build libraries if they are not available in the installation path.**

Start converting the project from ccsv3.3 to ccs5.2, please refer [chapter 2](#_Chapter_2) for conversion. Note that Chapter 2 refers old TI ARM7 library (*rts32tiabi.lib*). After building project as described in chapter 2 (**This is to make sure that there is no run time errors, please do not skip)**, right click on the Project and select properties.

Figure 3

Go to File search path under ARM Linker, and delete “{PROJECT\_ROOT}/../Linker Files/rts32tiabi.lib}” by selecting it and clicking on the button as directed by blue arrow in bellow picture (Figure 4)

Figure 4

Under *General* properties, select Run time support library as *<automatic>* from the drop down list.

And then select OK (Refer Figure 5)

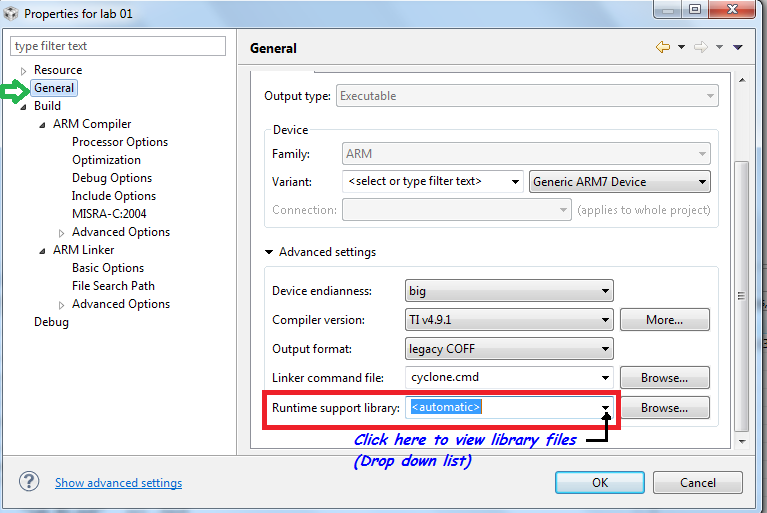
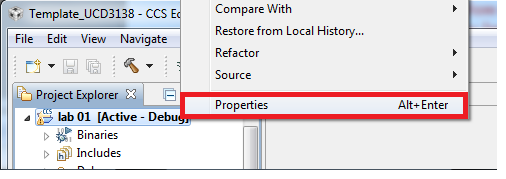
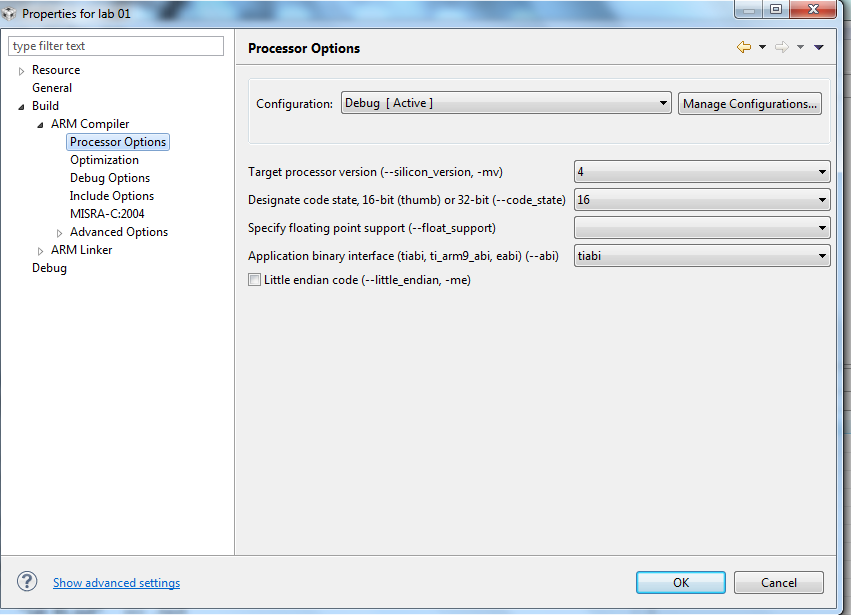


Figure 5

Again right click on the Project and select Properties.

Figure 6

Under Processor options select Application library Interface as **tiabi**. Refer bellow figure (Figure 7).

Figure 7

Right click on the Project and Select *Build Project. Please refer the following picture (Figure 8).*

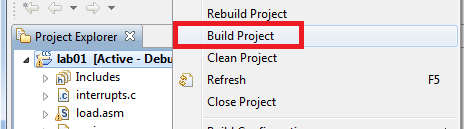


Figure 8

You will see the bellow picture (Figure 9) if you have console window visible.

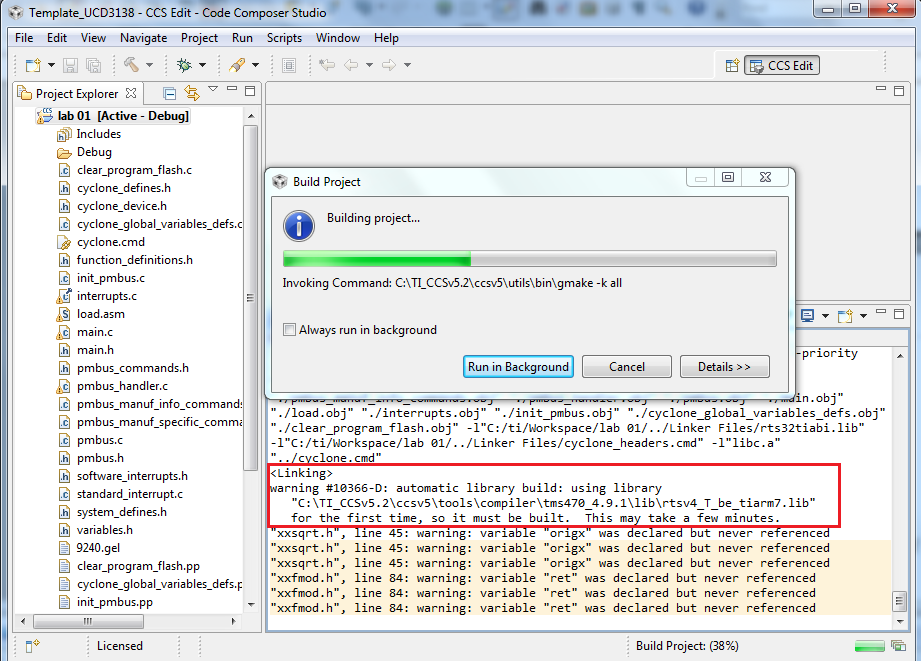


Figure 9

At First time this will take few minutes, wait till the build completes. Then you will see the new TI ARM7 library (rtsv4\_T\_be\_tiarm7.lib) in default library path of *tms470\_4.9.1* compiler. Refer the bellow Figure 10

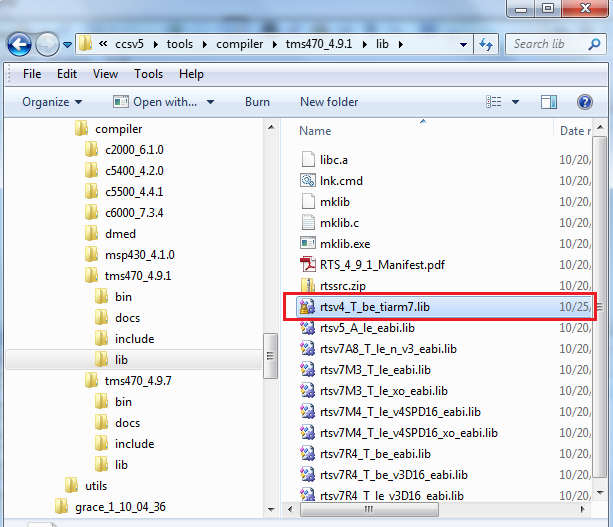


Figure 10.

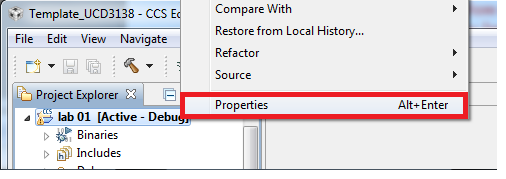
You have the new TI ARM7 library now!

**Step 3:** **Build Cyclone project using new TI ARM7 library (**rtsv4\_T\_be\_tiarm7.lib**).**

Start converting the a project from ccsv3.3 to ccs5.2, please refer chapter 2 for conversion (**Please come back here after done with Step 5 in Chapter 2**)

After the **Step 5** in the chapter 2,

Right click on the project and select properties.

Figure 11

Go to File search path under ARM Linker, and delete “{PROJECT\_ROOT}/../Linker Files/rts32tiabi.lib}” by selecting it and clicking on the button as directed by blue arrow in bellow picture.

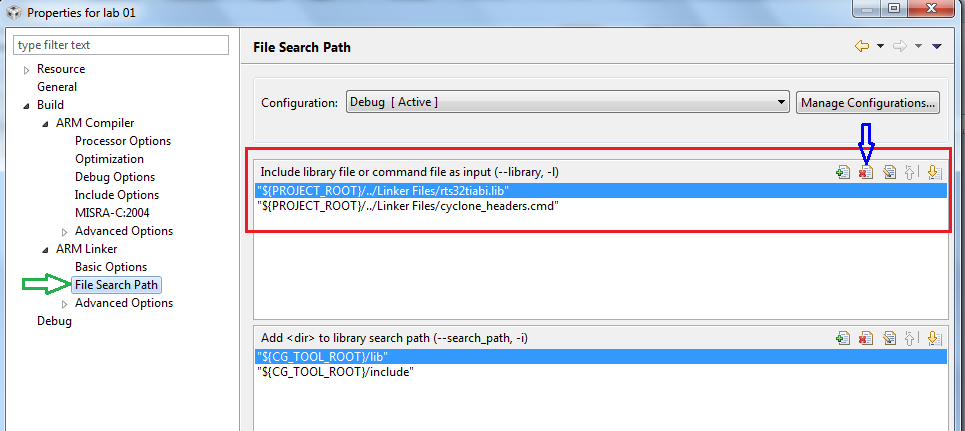


Figure 12

Under *General* properties, select Run time support library as ***<rtsv4\_T\_be\_tiarm7.lib>*** from the drop down list (refer Figure 13).

And then select OK.

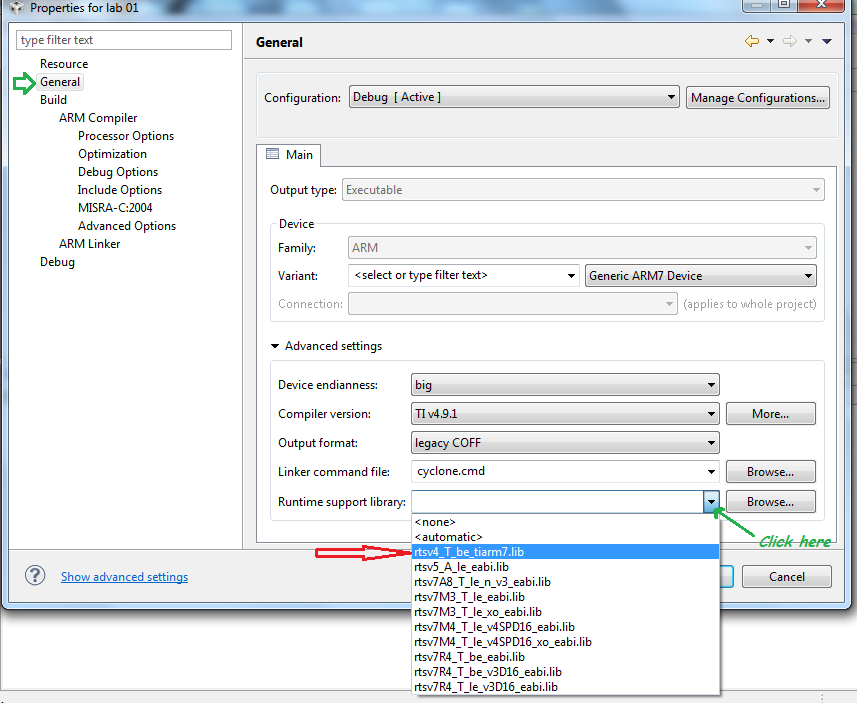
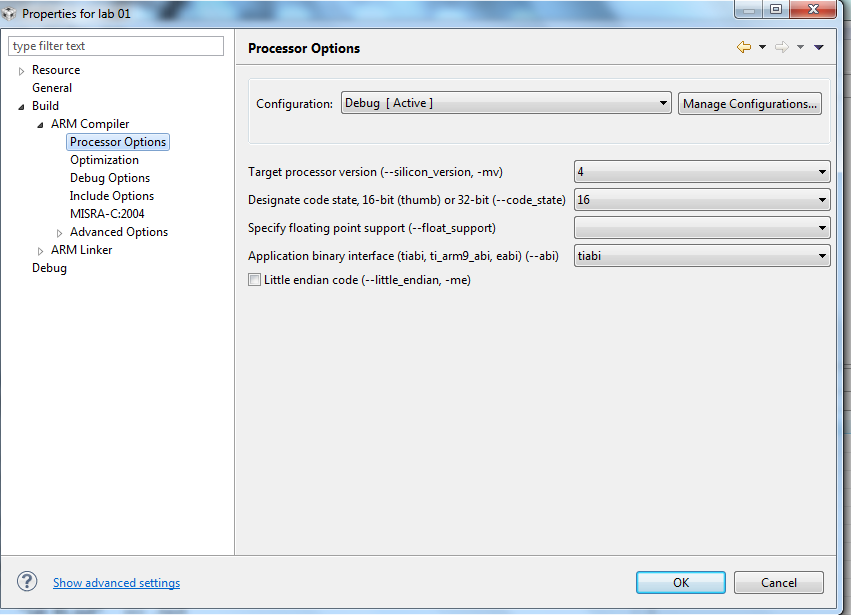


Figure 13

Then under *Processor options* select *Application library Interface* as **tiabi**. Refer bellow figure 14.

Figure 14

Right click on the Project and Select *Build Project. Please refer the following picture(refer Figure 15).*

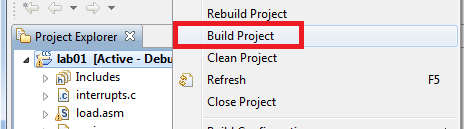


Figure 15