Standard Product

Creating UT32M0R500 Project in the Keil IDE

UT32M0R500 32-bit Arm® Cortex® M0+ Microcontroller

Application Note

Cobham.com/HiRel

February 23, 2018

The most important thing we build is trust

Table 1: Cross Reference of Applicable Products

PRODUCT NAME	MANUFACTURER PART NUMBER	SMD #	DEVICE TYPE	INTERNAL PIC NUMBER
Arm Cortex M0+	UT32M0R500	5962-17212	Project Setup	QS30

1.0 OVERVIEW

This document details the process of creating a Wolverine-based embedded software project using the Keil ARM development tools. For the purposes of this document, we will create a project named MyProjectA and configure the Keil tools to include all the source modules required for a successful build. Using this template, the user should be able to create projects using (a) their preferred application source directory structures and (b) the directory structure for the Keil-supplied files.

2.0 PROJECT CREATING

2.1 Step #1: Create the Develop Directories

Our suggested method for creating a Wolverine software development project begins with creating three directories on your development computer:

- Keil_v**X**
- Wolverine
- Project(s)

The Keil_v**X** directory will be created when you install your Keil development tools. [The '**X**' will be determined by the version of Keil tools you are using.] The Keil integrated development environment (IDE) is named µVision.

The Wolverine directory will contain the Cobham-provided, Wolverine-specific files, including the Standard Peripheral Library files. The Wolverine directory structure will look like this:

```
\UT32M0R500_API_vX_X_X
 *-> \StdPeriphLib
      *-> \src
           ** contains: ut32m0_XYZ.c [numerous files]
      *-> \inc
           ** contains: ut32m0_XYZ.h [numerous files]
 *-> \UT32M0SpecificARM
      ** contains: UT32M0_SRAM_Debug.ini
      *-> \src
           ** contains: startup_ARMCM0plus.s, system_ARMCM0plus.c
      *-> \inc
           ** contains: UT32M0R500.h
      *-> \SVD
           ** contains: UT32M0R500_BasiCAN.SFR, UT32M0R500_PeliCAN.SFR
```

The Project directory will contain (a) your project's source files and (b) the project files created by the Keil development tools. **As you create additional projects, only this step needs to be repeated.**

Create the `MyProjectA` directory and two sub-directories under that: `ApplicationFiles` and `ProjectFiles`.

```
\MyProjectA
*-> \ApplicationFiles
      *-> \ABC   [organize these sub-directories...
      *-> \DEF   [...as best suits your project]
      *-> ...
*-> \ProjectFiles
      *-> ...    [Keil's project creation tool will create...
      *-> ...    ...this sub-directory structure.]
```

Move your application files to the `\MyProjectA\ApplicationFiles` directory.

2.2 Step #2: Create Your Project

Our next step is to create the project within the Keil development. To begin, open Keil's µVision and…

- Under the 'Project' pull-down, select 'New µVision Project'.
- Enter your project name in the 'File name' box. In this case, we'll use "MyProjectA".
- Use the directory navigator to find and highlight the sub-directory `MyProjectA\ProjectFiles`.
- Press the 'Open' button. Then press the 'Save' button. The 'Select Device…' window will appear.

Then:

- In the 'Select Device…' window, expand the 'ARM' selection and expand the 'ARM Cortex M0 plus' selection.
- Select 'ARMCM0P'.
- Press the 'OK' button. The 'Manage Run-Time Environment' window will appear.

Then:

- In the 'Manage Run-Time Environment', press the 'OK' button.

To verify that the project was created, use Windows Explorer (external application) to navigate to the `MyProjectA\ProjectFiles` directory. You should see two files (MyProjectA.uvoptx and MyProjectA.uvprojx) and two sub-directories (Listings and Objects). If these are present, the project was successfully created.

2.3 Step #3: Add the "Standard Peripheral Library" Files to Your Project

In µVision...

First:

- In the 'Project' panel, right-click 'Target 1' and select 'Add Group'. A new group will appear named "New Group".
- Highlight (single left-click) "New Group" and rename it **"StdPeriphLib_src"**. Press the 'enter' key.
- Right-click "StdPeriphLib_src" and select 'Add existing files…' A directory navigation window will appear.
- Navigate to the `\UT32M0R500_API_vX_X_X\StdPeriphLib\src` directory.
- Select 'Files of type' to be **'C Source file (*.c)'** and select all of the `ut32m0_XYZ.c` files.
- Press the 'Add' button. Close the window with the 'Close' button if it doesn't close by itself.

Then:

- In the 'Project' panel, right-click 'Target 1' and select 'Add Group'. A new group will appear named "New Group".

- Highlight (single left-click) "New Group" and rename it **"StdPeriphLib_inc"**. Press the 'enter' key.
- Right-click "StdPeriphLib_inc" and select 'Add existing files…' A directory navigation window will appear.
- Navigate to the `\UT32M0R500_API_vX_X_X\StdPeriphLib\inc` directory.
- Select 'Files of type' to be **'Text file (*.txt; *.h; *.inc)'** and select all of the `ut32m0_XYZ.h` files.
- Press the 'Add' button. Close the window with the 'Close' button if it doesn't close by itself.

2.4 Step #4: Add the Wolverine-Specific ARM Files to Your Project

In µVision...

First:

- In the 'Project' panel, right-click 'Target 1' and select 'Add Group'. A new group will appear named "New Group".
- Highlight (single left-click) "New Group" and rename it **"UT32M0SpecARM"**. Press the 'enter' key.
- Right-click "WolvSpecARM" and select 'Add existing files…' A directory navigation window will appear.
- Navigate to the `\UT32M0R500_API_vX_X_X\UT32M0SpecificARM\src` directory.
- Select 'Files of type' to be **'All files (*.*)'** and select the **`startup_ARMCM0plus.s`** and **`system_ARMCM0plus.c`** files.
- Press the 'Add' button. Close the window with the 'Close' button if it doesn't close by itself.

Then:

- Right-click "UT32M0SpecARM" and select 'Add existing files…' A directory navigation window will appear.
- Navigate to the `\UT32M0R500_API_vX_X_X\UT32M0SpecificARM\inc` directory.
- Select 'Files of type' to **'Text file (*.txt; *.h; *.inc)'** and select the **`UT32M0R500.h`** file.
- Press the 'Add' button. Close the window with the 'Close' button if it doesn't close by itself.

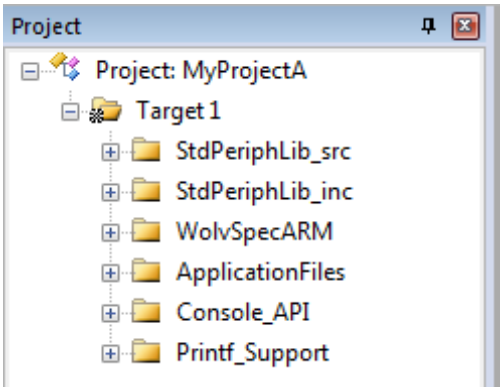
2.5 Step #5: Add Your Application Files to Your Project

In µVision...

- In the 'Project' panel, right-click 'Target 1' and select 'Add Group'. A new group will appear named "New Group".
- Highlight (single left-click) "New Group" and rename it **"ApplicationFiles"**. Press the 'enter' key.
- Right-click "ApplicationFiles" and select 'Add existing files…' A directory navigation window will appear.
- Navigate to the `MyProjectA\ApplicationFiles` directory.
- Select 'Files of type' to be **'All files (*.*)'** and select all the files you wish to add to the project.
- Press the 'Add' button. Close the window with the 'Close' button if it doesn't close by itself.

Note: You may wish to organize your project files differently under 'Target 1' within the 'Project' panel. Exercising this option is entirely at your discretion. If you have been provided with Cobham-sourced **Console_API** and **Printf_Support** file sets, add them to the project as you would your application files.

The 'Project' panel should now appear as follows:

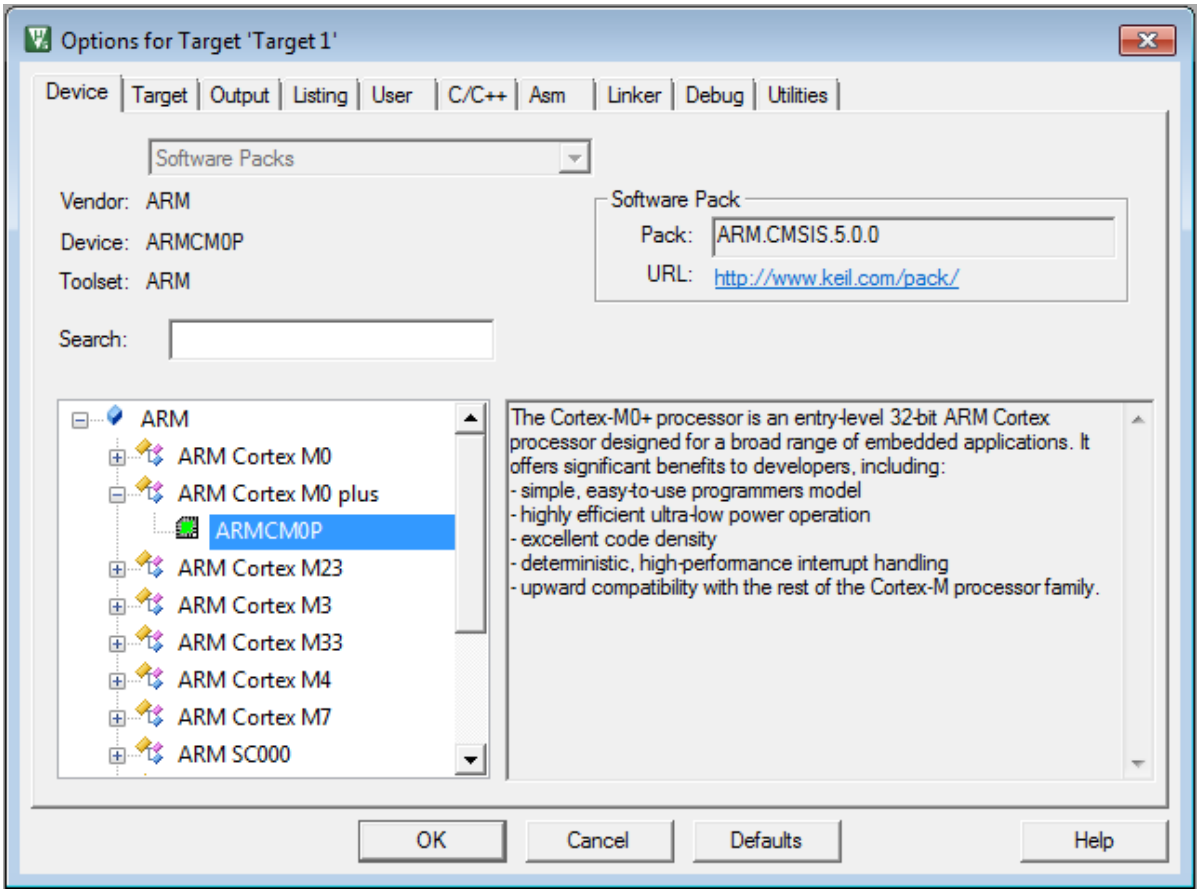


2.6 Step #6: Configure the Project

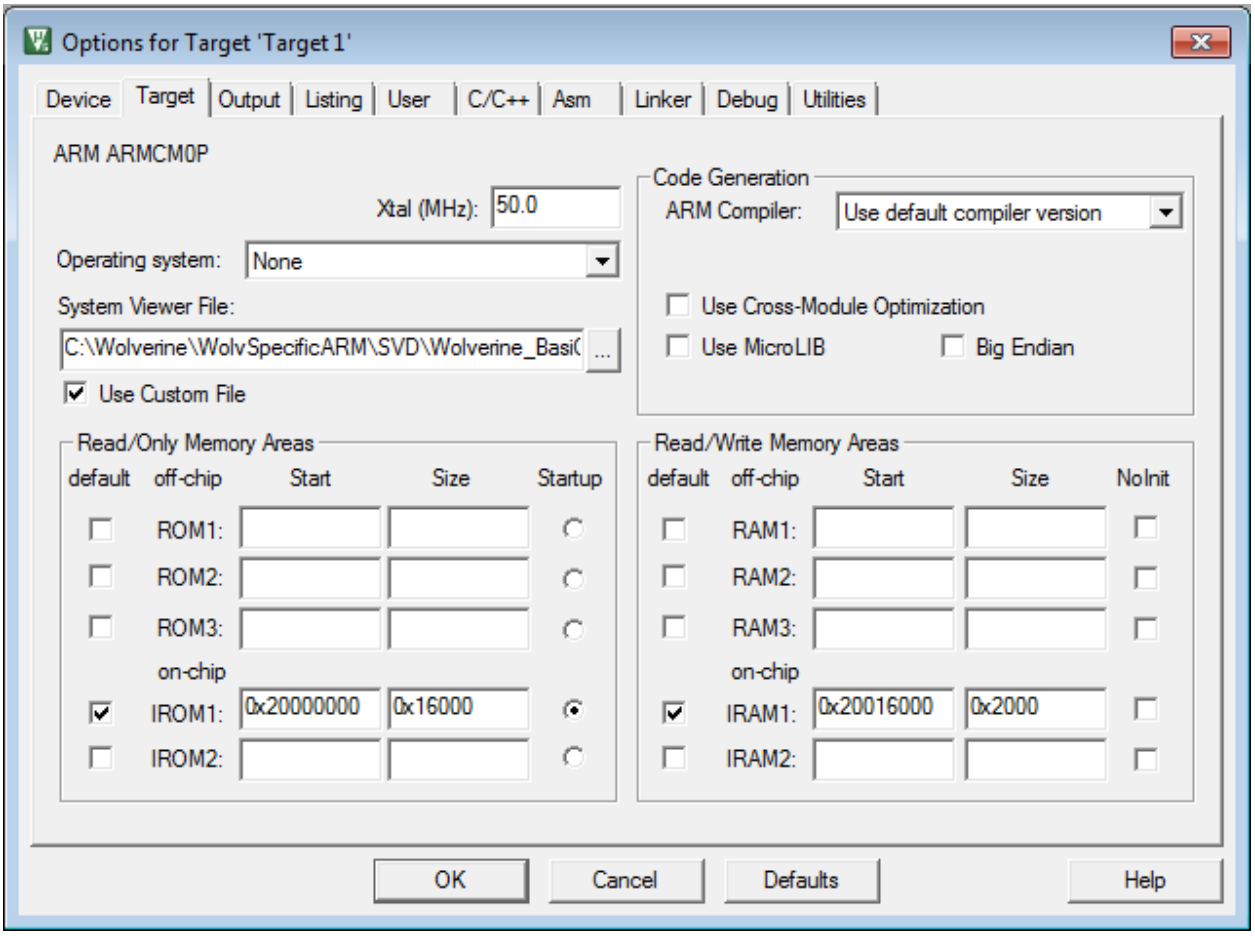
In µVision, in the **'Project'** panel, right-click on 'Target 1' and select **'Options for Target 'Target 1'...'**. Configure the settings according to the following screenshots:

(Note: Your settings for 'Software Pack' will be dependent on what pack you have installed for the Keil tools.)

Select the **'Device'** tab and configure as:



Select the **'Target'** tab and configure as:



Note that the **'System Viewer File'** is:

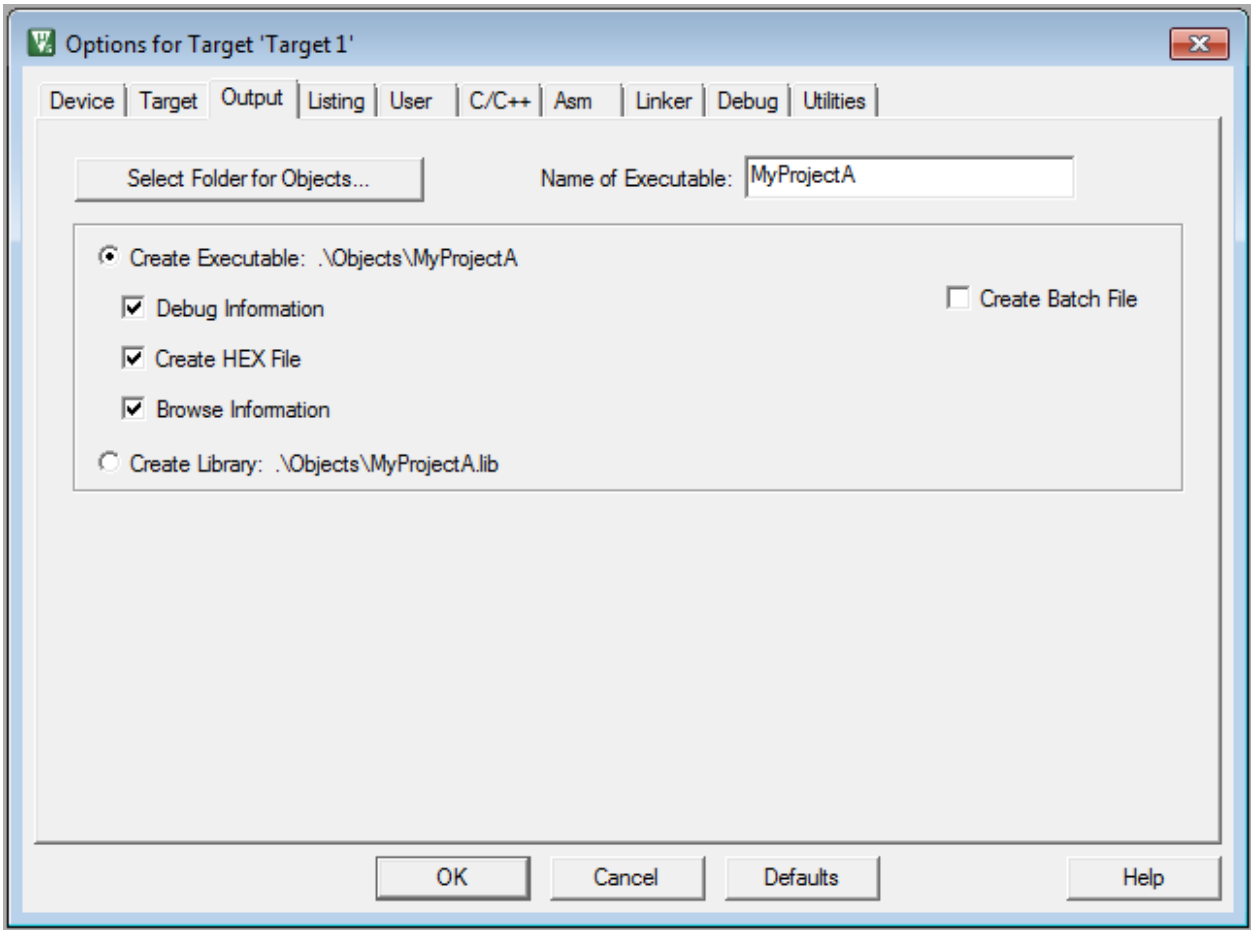
`\UT32M0SpecificARM\SVD\UT32M0R500_BasiCAN.SFR`

If your project requires the use of PeliCAN, set the path to:

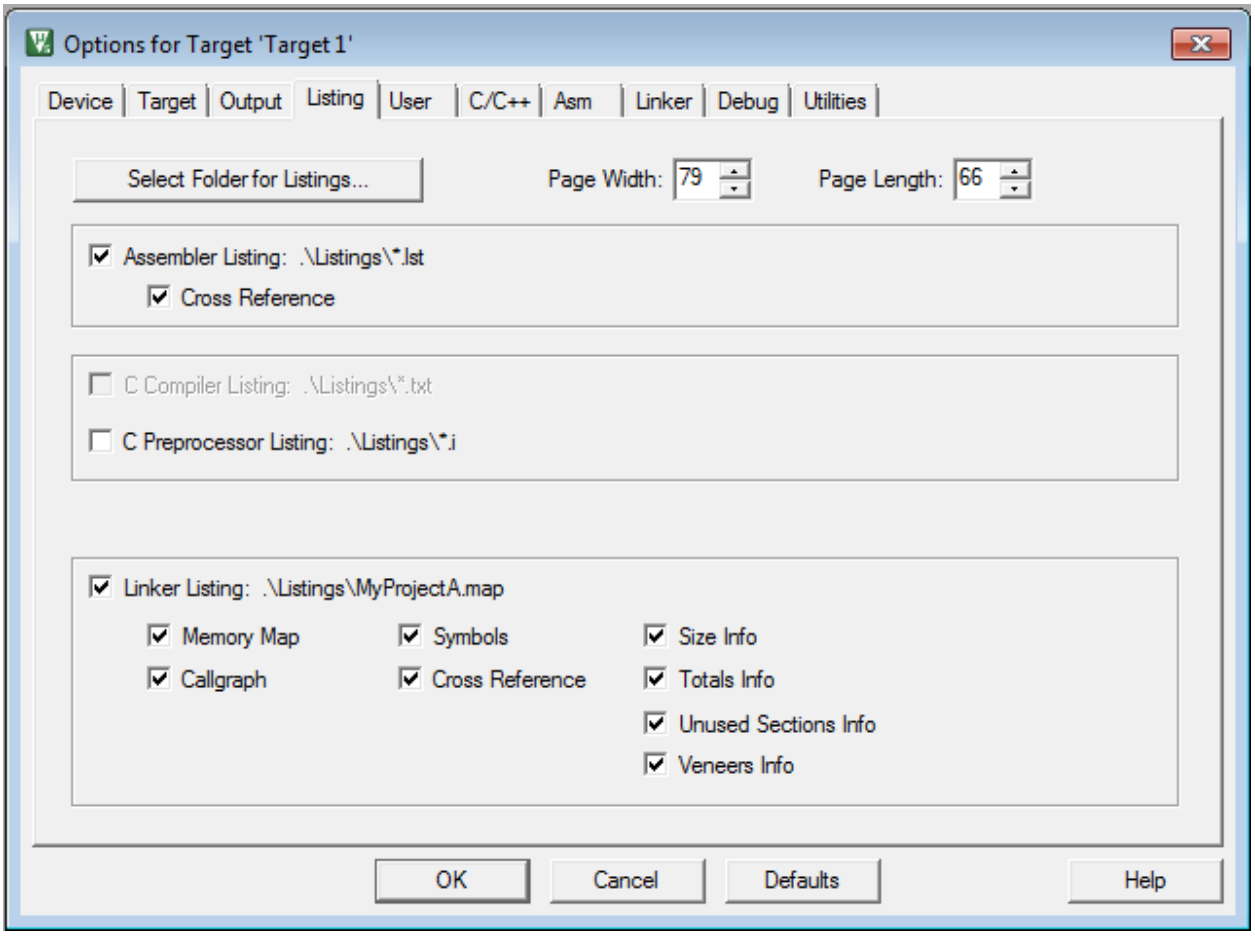
`\UT32M0SpecificARM\SVD\UT32M0R500_PeliCAN.SFR`

The actual location is based on system installation as done in Step #1.

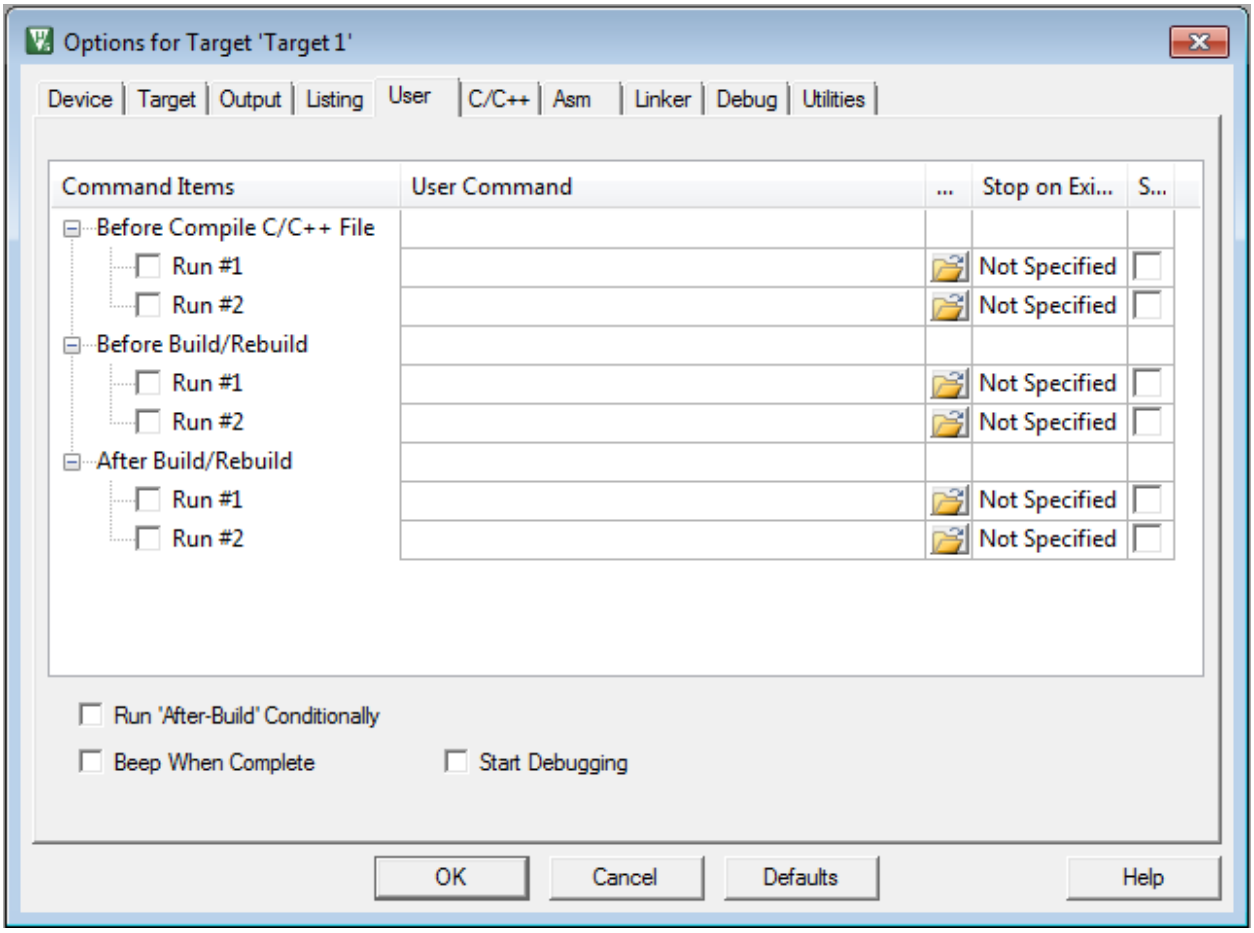
Select the **'Output'** tab and configure as:



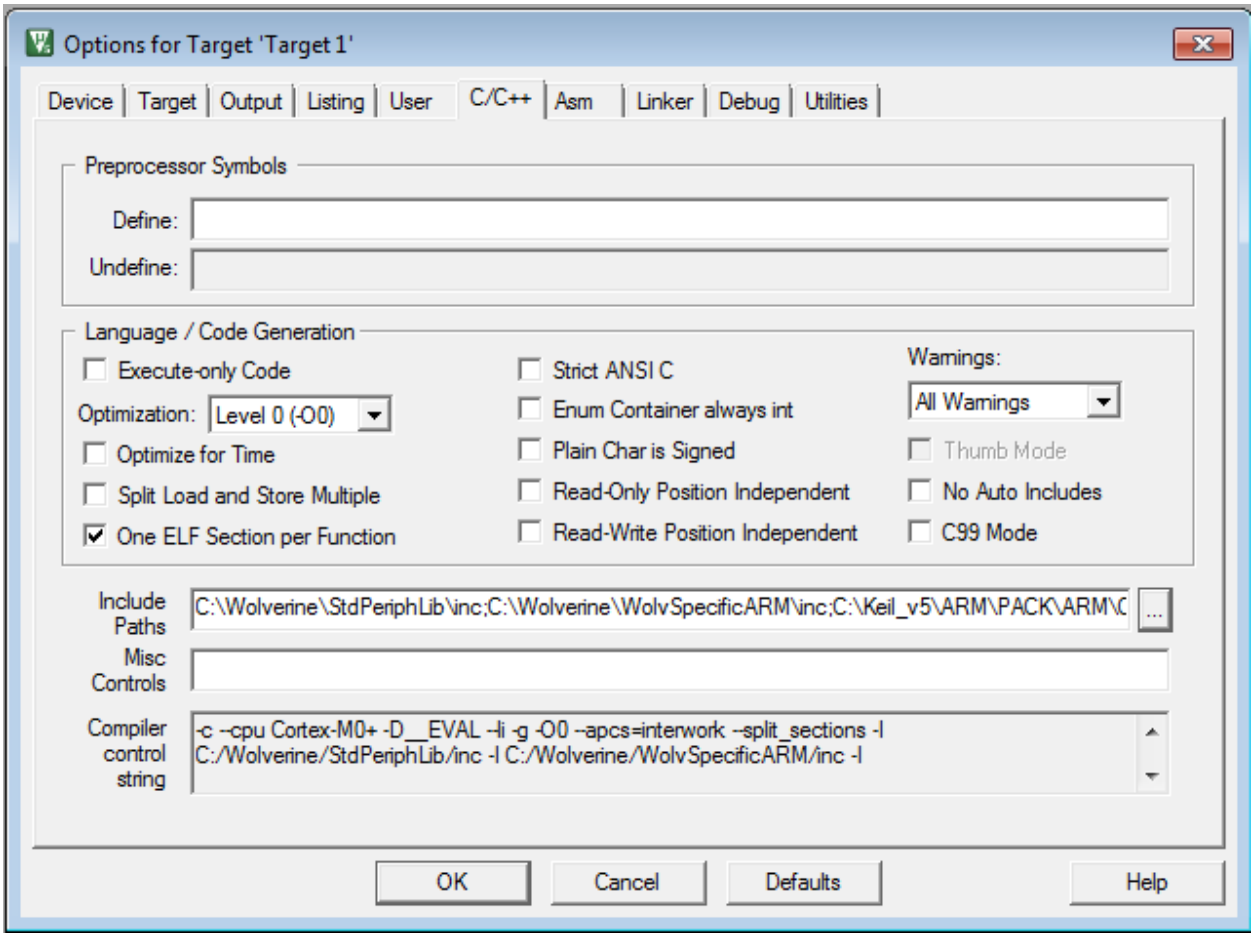
Select the **'Listing'** tab and configure as:



Select the **'User'** tab and configure as:



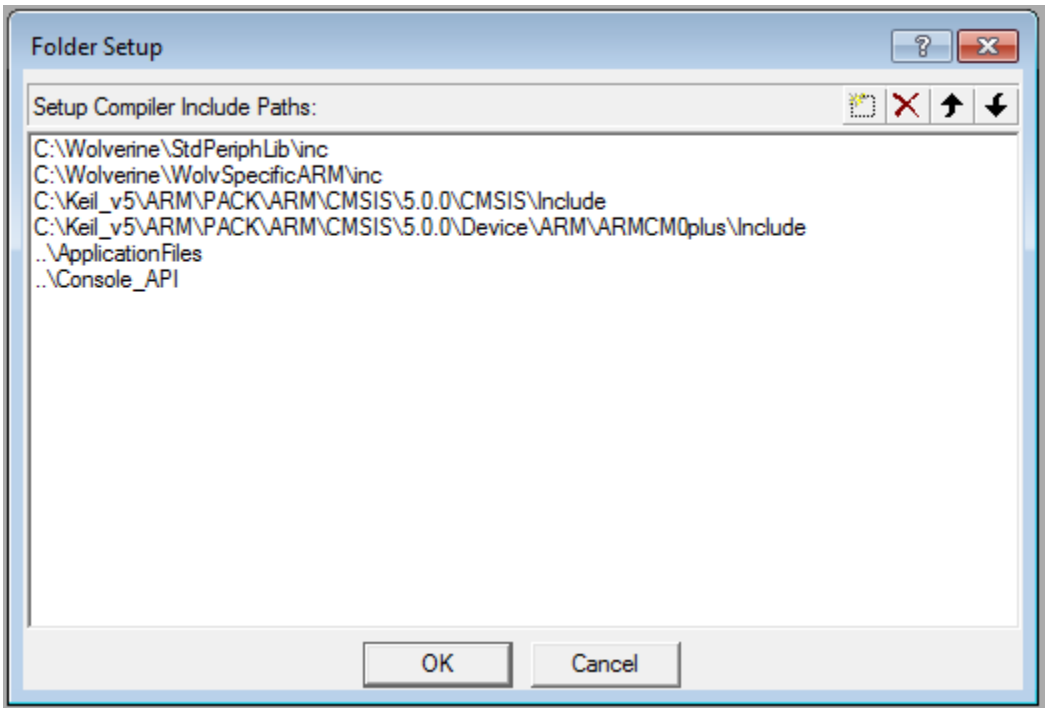
Select the **'C/C++'** tab and configure as:



Note that if your project requires the use of PeliCAN, include **`CAN_ENABLE_PELICAN_SUPPORT`** in the **'Processor Symbols' : 'Define'** text box.

Press the **'Include Paths'** navigator tab ('…') to get the following window…

… and configure as:



Note that the `\Wolverine` and `\WolvSpecificARM` directories refer to the `\UT32M0R500_API_vX_X_X` and `\UT32M0SpecificARM` directories.

Note that the two Keil directories…

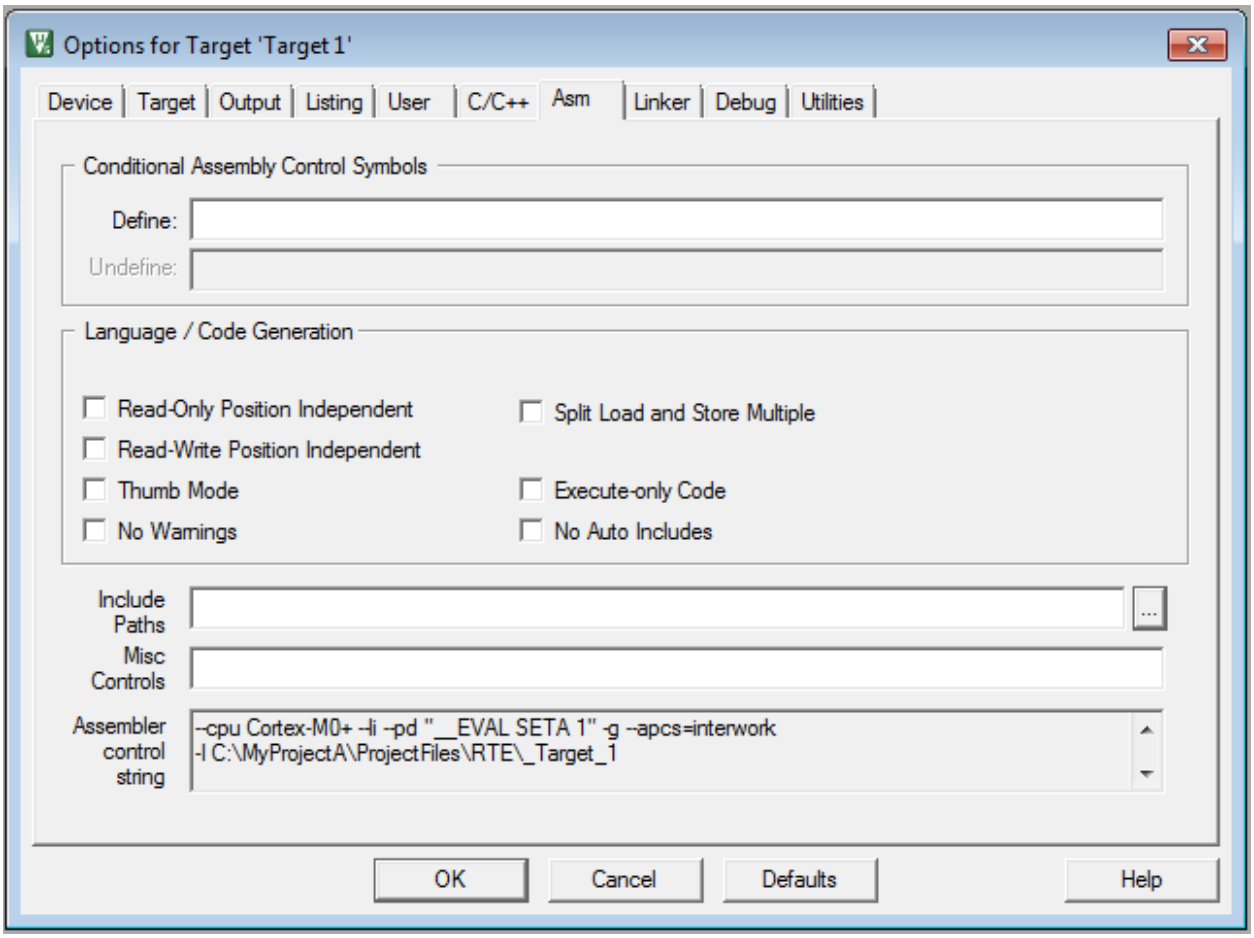
`C:\Keil_v5\ARM\PACK\ARM\CMSIS\5.0.0\CMSIS\Include`
`C:\Keil_v5\ARM\PACK\ARM\CMSIS\5.0.0\Device\ARM\ARMCM0plus\Include`

…may have different path names based on tool and pack version numbers (in **red**).

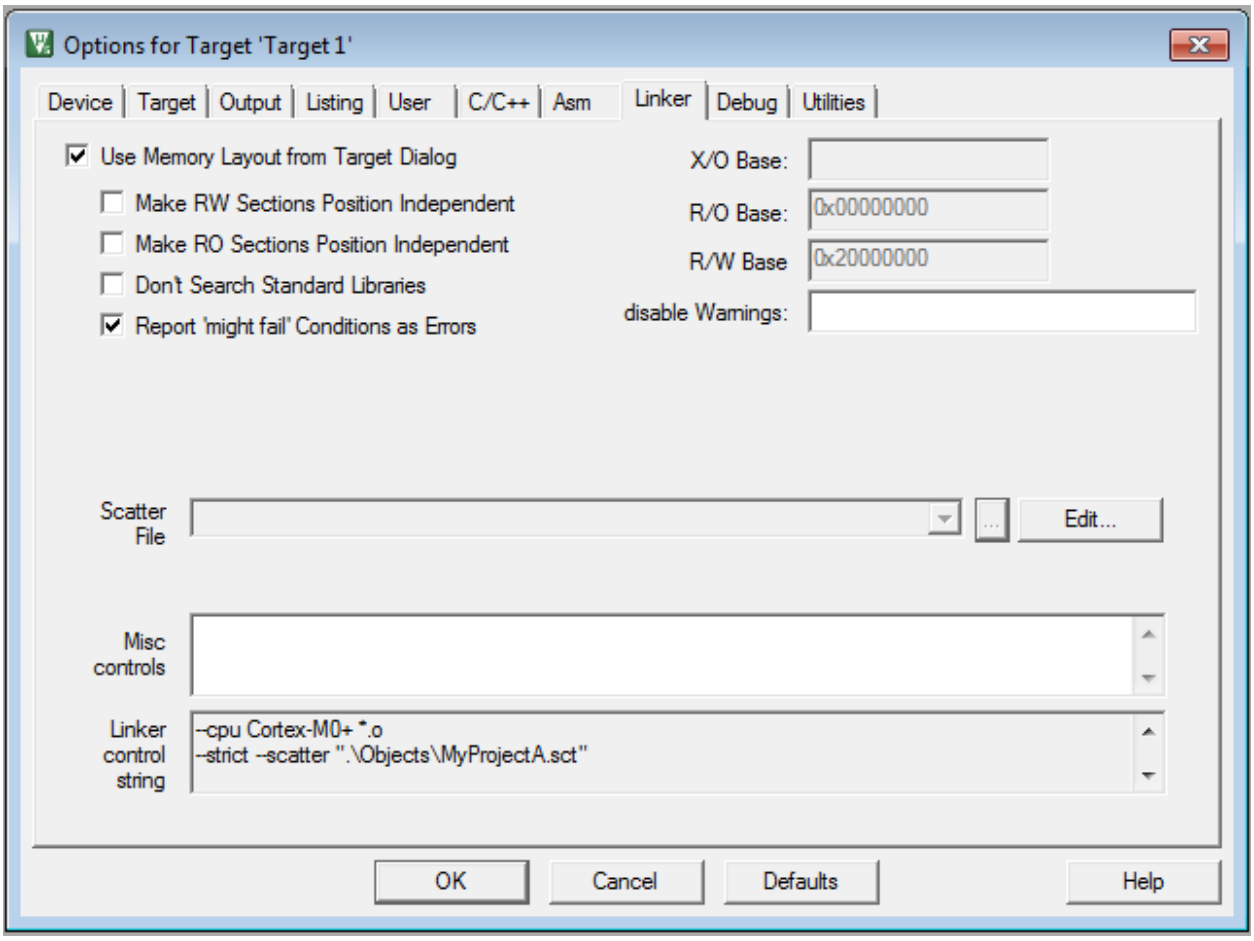
Note that the StdPeriphLib\inc and UT32M0SpecificARM\inc directories may have different path names based on system installation performed in Step #1.

Press the **'OK'** button to save the settings.

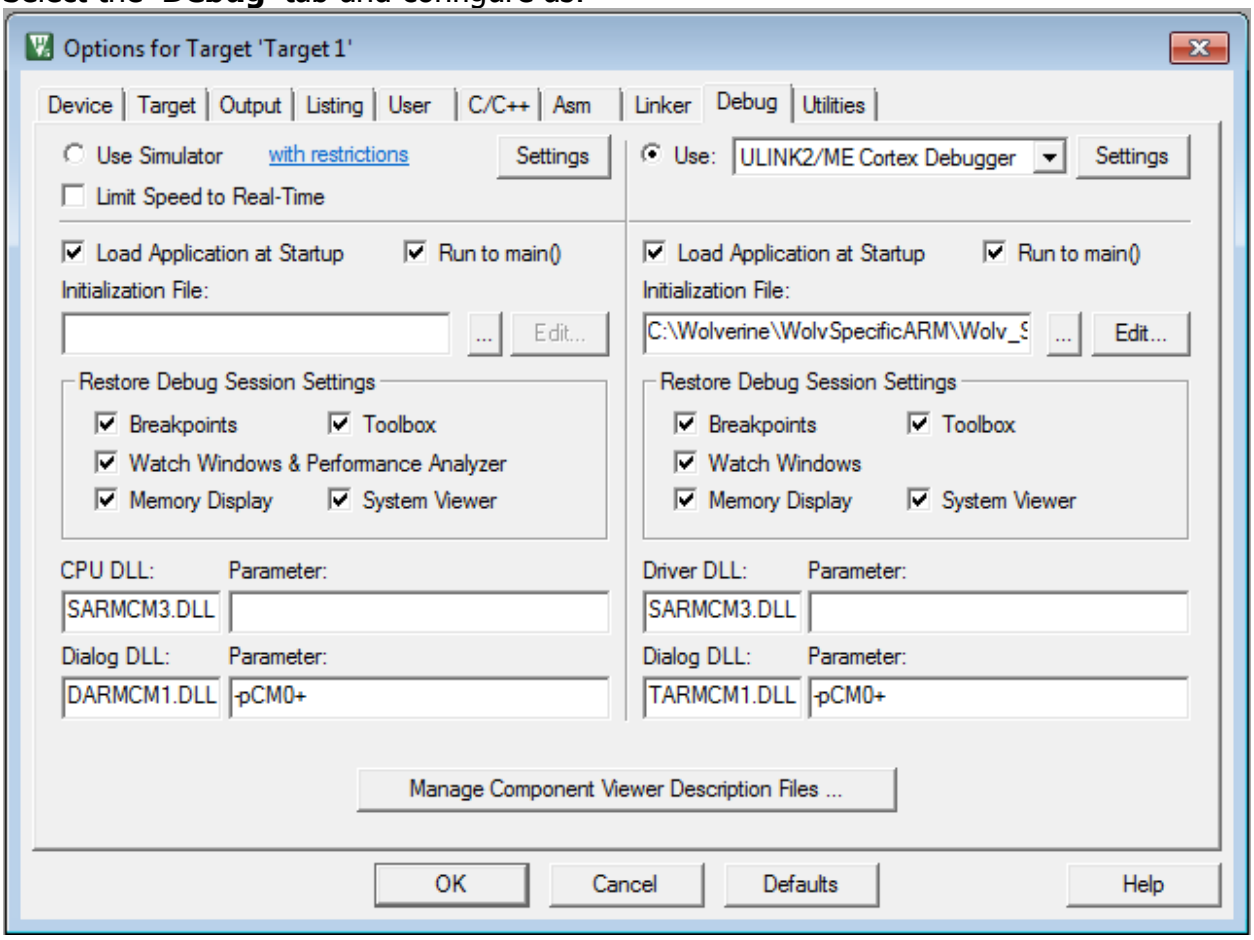
Select the **'Asm'** tab and configure as:



Select the **'Linker'** tab and configure as:



Select the **'Debug'** tab and configure as:

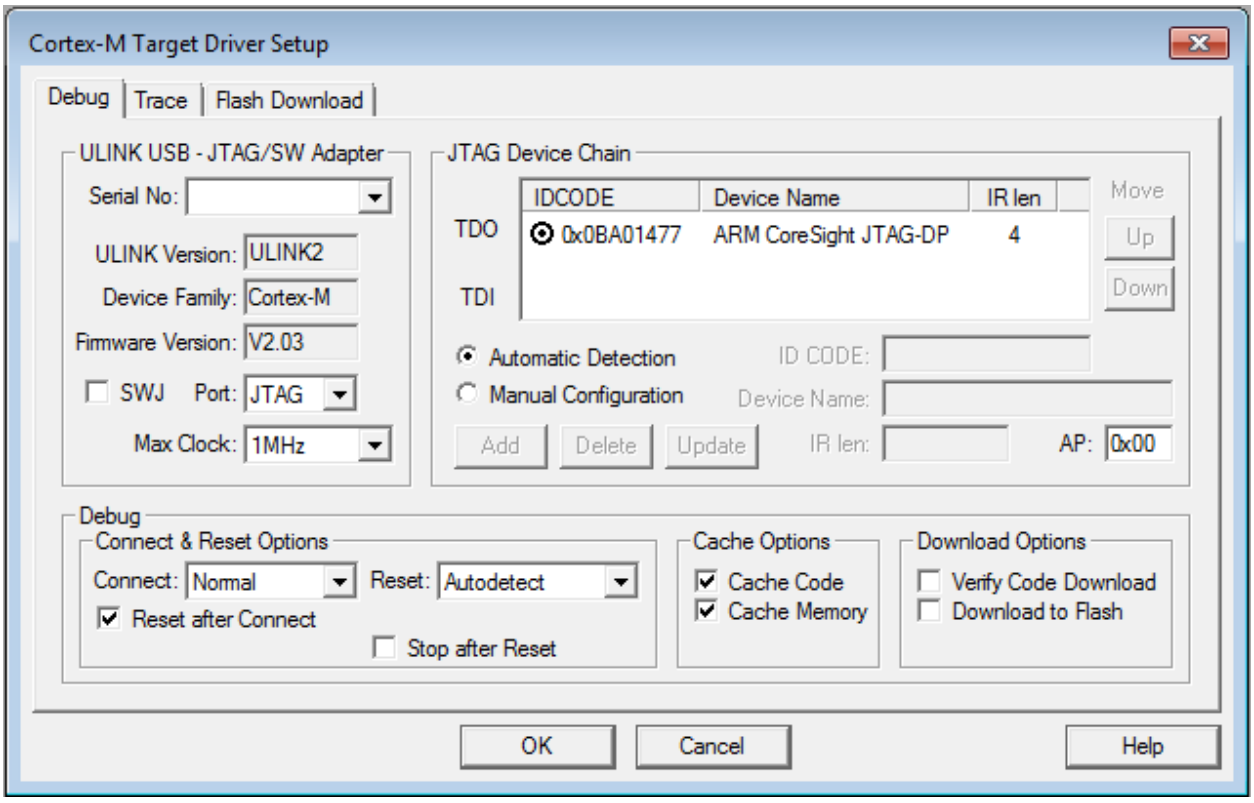


Note that the full **'Initialization File'** path is:

`C:\UT32M0R500_API_vX_X_X\UT32M0SpecificARM\UT32M0_SRAM_Debug.ini`

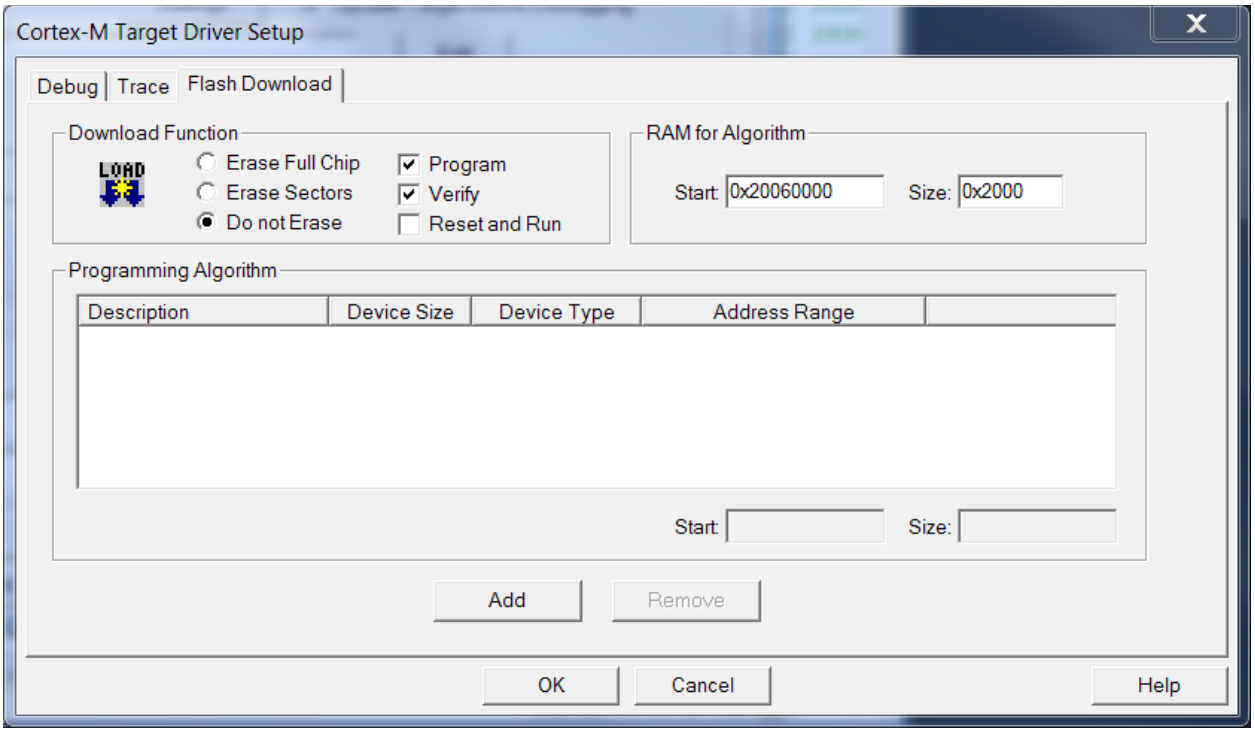
Press the (debugger) **'Settings'** button and the 'Target Driver Setup' window will appear.

Under the **'Debug'** tab, the following configurations are desired:



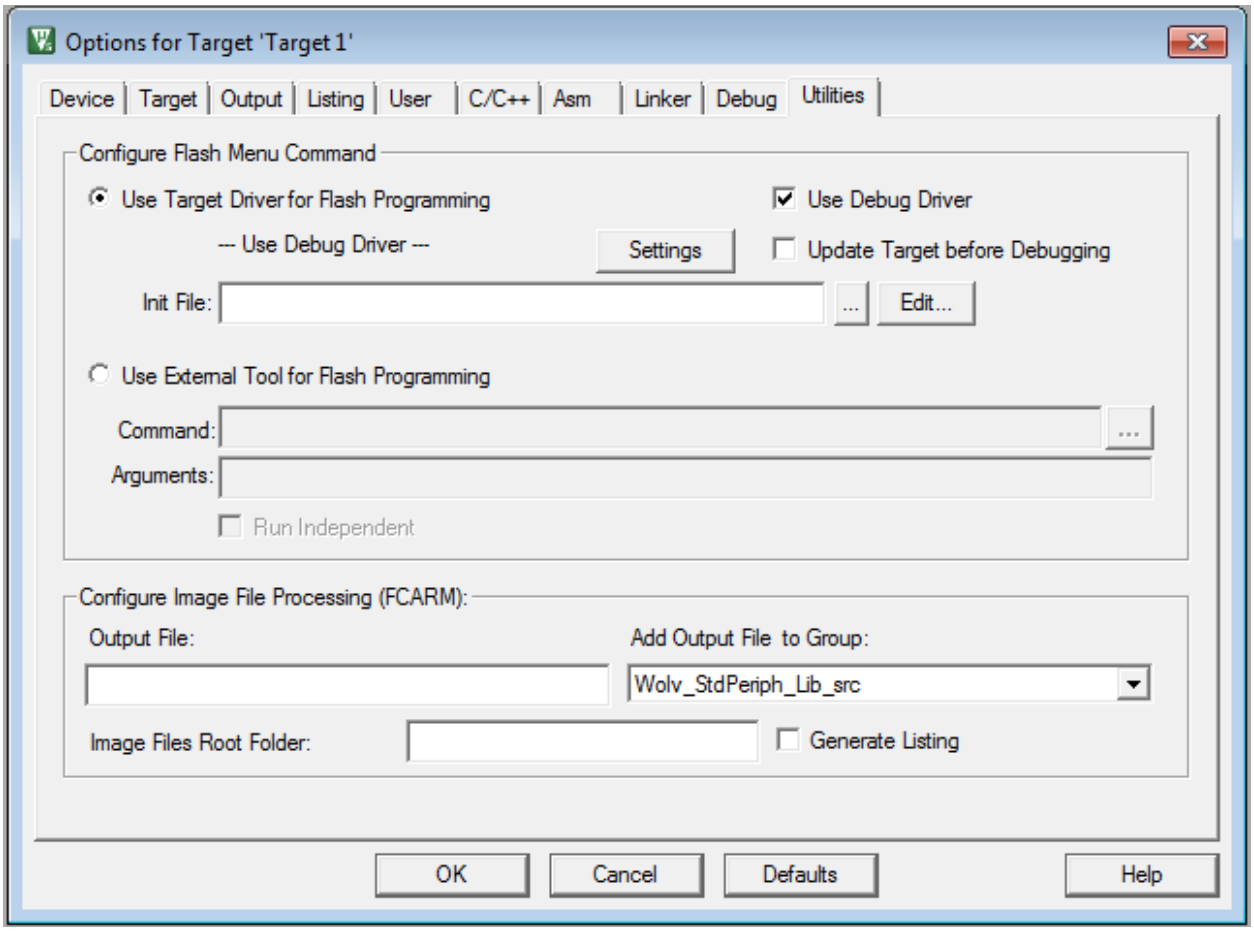
Note that the serial for your JTAG pod will appear in the 'Serial No' box.

Under the **'Flash Download'** tab, the following settings are desired:



Press the **'OK'** button to save the settings.

Back to the project **'Options'** window, set the following settings under the **'Utilities'** tab.



Press the 'OK' button to save the settings.

2.7 Step #7: Building Your Project

Your project is now ready to be built (compiled/assembled/linked).

Press the **'Build'** button, the **'Rebuild'** button…

…or press **'F7'** on the keyboard to build the project. The progress of the build – including any warnings or errors – will be displayed in the **'Build Output'** panel. Doubling-clicking any error or warning will activate the source code editor and place the cursor on the line of code that created the build issue.

2.8 Step #8: Executing / Debugging Your Project

Once there's been a successful build of the project, it is ready for executing and/or debugging.

To start the session, press the debug button in the top toolbar…

…or press **'Ctrl+F5'** on the keyboard. This will activate the debugger, load the image into Wolverine's SRAM, and run to **`main()`**.

The application can be (free-)run by pressing the run button…

…or by pressing **'F5'** on the keyboard.

The application can be 'stepped' by using any of the four step buttons:

The application can be stopped by pressing the stop button:

For additional features of the debugger – including the setting/clearing of breakpoints – refer to the Keil documentation.

3.0 REVISION HISTORY

Date	Rev. #	Author	Change Description
May 2017	1.0.0	SW	Initial Release
Dec 2017	1.0.1	AW	Minor edits for directory names
Feb 2018	1.0.2	AW	Additional edits for directory names and dialog settings box

Cobham Semiconductor Solutions

This product is controlled for export under the U.S. Department of Commerce (DoC). A license may be required prior to the export of this product from the United States.

Cobham Semiconductor Solutions
4350 Centennial Blvd
Colorado Springs, CO 80907

E: info-ams@aeroflex.com
T: 800 645 8862