Stellaris® Development and Evaluation Kits for Keil™ RealView® MDK

The Stellaris Development and Evaluation Kits provide a low-cost way to start designing with Stellaris microcontrollers using Keil's RealView Microcontroller Development Kit (MDK) for ARM controllers. The boards can function as either a complete evaluation target or as a debugger interface to any external Stellaris device.

Requirements

- You have a PC with a USB interface, running Microsoft® Windows 2000, XP, or Vista
- You have the Stellaris Evaluation Kit Documentation and Software CD or the standalone ARM/Keil CD found in the Development Kit

CAUTION: There is a known electrical issue with the FT2232 device that is used in the on-board In Circuit Debug Interface (ICDI). Some USB hubs can cause the device to misbehave, with symptoms ranging from failed enumeration to corrupt data transfers. If you experience trouble when using the on-board ICDI, try connecting the USB cable directly to one of the USB ports on your PC or laptop.

Keil™ RealView® MDK

This quickstart shows you how to install the evaluation version of the Keil RealView MDK, and then how to use it to build and run an example application on your Stellaris Evaluation Board.

Step 1: Install the RealView MDK Tools

1. Insert the Evaluation Kit Documentation and Software CD or the standalone ARM/Keil CD into the CD-ROM drive of your computer. If Autoplay is enabled on your PC, the index.htm file automatically opens in your default web browser. If not, use Windows Explorer to open it manually.

2. With the Evaluation Kit CD, click the Tools button and then the Keil Logo to download a zip file containing the installer. If you prefer to run the setup executable from the CD, use Windows Explorer to view the files on the CD and double-click the MDKxxx.exe file located in the Tools\Keil\ directory.

   With the standalone CD, follow the installer dialog.

Step 2: Install the StellarisWare® Package

A full set of C-based peripheral drivers is provided, covering all peripherals and functionality of the Stellaris devices. The StellarisWare package includes various example applications with project files for all major tool vendors that support Stellaris, including Keil. To install StellarisWare, follow these steps:

1. Navigate to the *Tools* tab on the Evaluation Kit Documentation and Software CD, or to the *Software* tab on the Development Kit Documentation and Software CD.

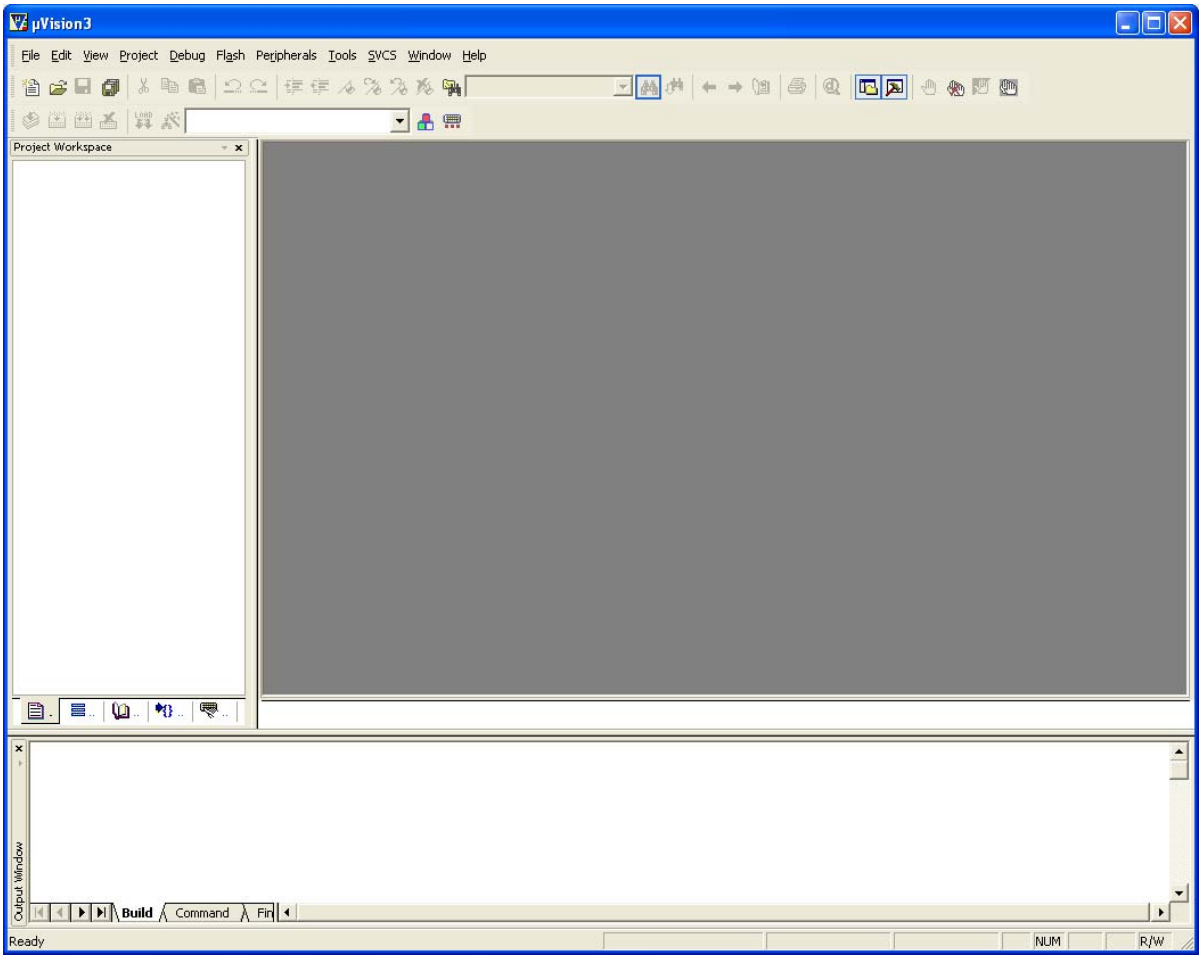
   NOTE: If you are navigating the CD using Windows Explorer (or a similar application), go to the Tools/StellarisWare or Software/StellarisWare directories.

2. Click on the 'Install' link next in the StellarisWare section (under Tools or Software) of the CD and run the StellarisWare installer. If you prefer to manually install StellarisWare, the installer is a self-extracting zip file that is located in the Tools/StellarisWare directory. You can use a zip file extraction utility such as WinZip to manually extract the contents.

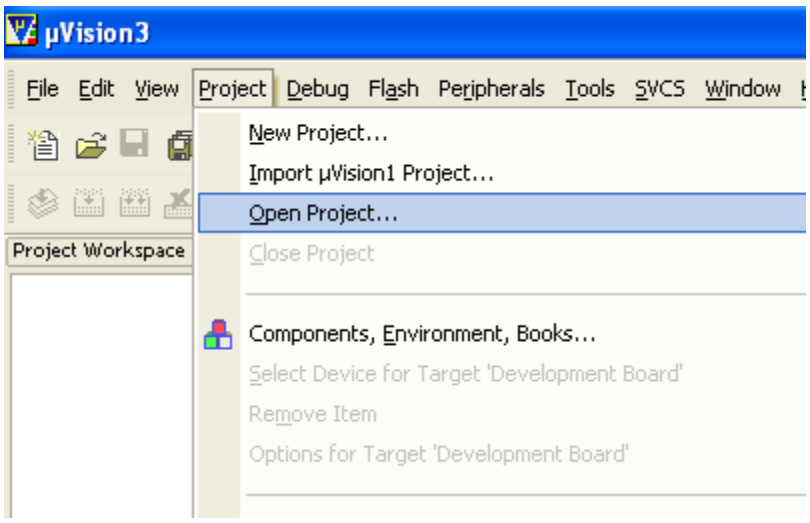
3. To view the StellarisWare documentation, navigate to the installation directory and click on the *Stellaris Peripheral Driver Library User's Guide* PDF.

Step 3: Start the Keil µVision IDE and Load the Hello Project

1. Start the Keil µVision IDE by double-clicking the icon on your desktop or by selecting it from the Windows Start Menu.



2. From the Project menu, select “Open Project.”



3. Use the dialog box to navigate to the Hello program in the directory appropriate for your board. From the location where you installed StellarisWare, the Hello project is located in:

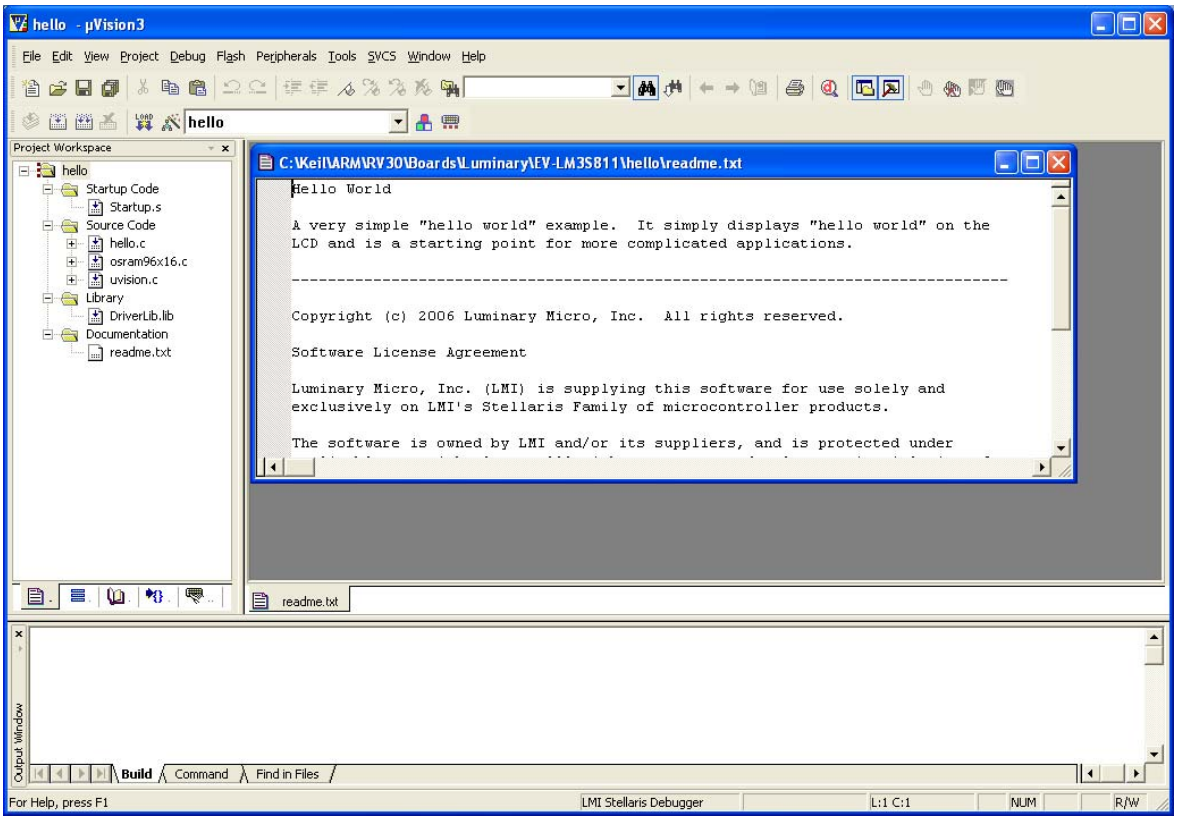
   ```
   StellarisWare\boards\{board_name}\hello
   ```

   The Keil tools also install StellarisWare as part of the default installation, but the version may be older than what is currently available in the development or evaluation kit. You can find StellarisWare in the Keil tree by looking in:

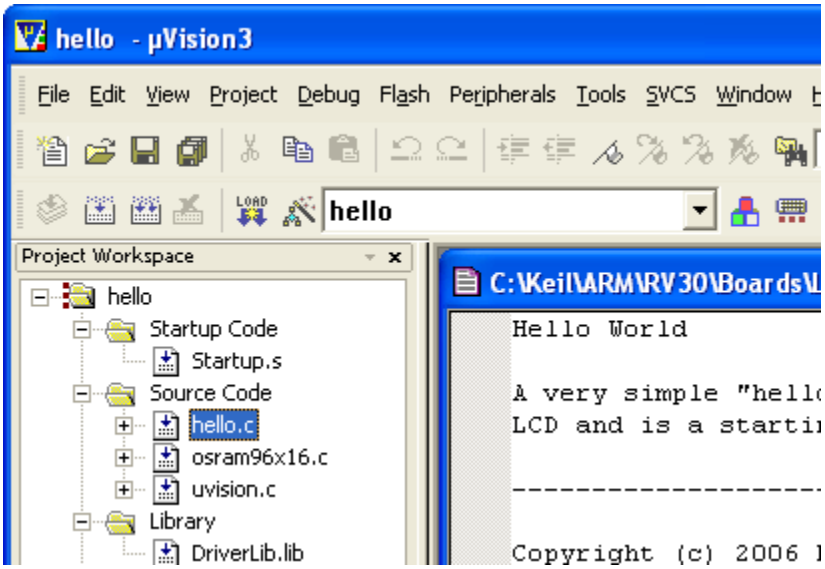
   ```
   C:\Keil\ARM\Boards\Luminary\{board_name}
   ```

Important: For the most recent version of the StellarisWare example projects, use the standalone version found on the Evaluation Kit Documentation and Software CD or check www.ti.com/stellaris for the latest software updates.

4. Select the hello.uvproj project file and click Open. The project opens in the IDE.

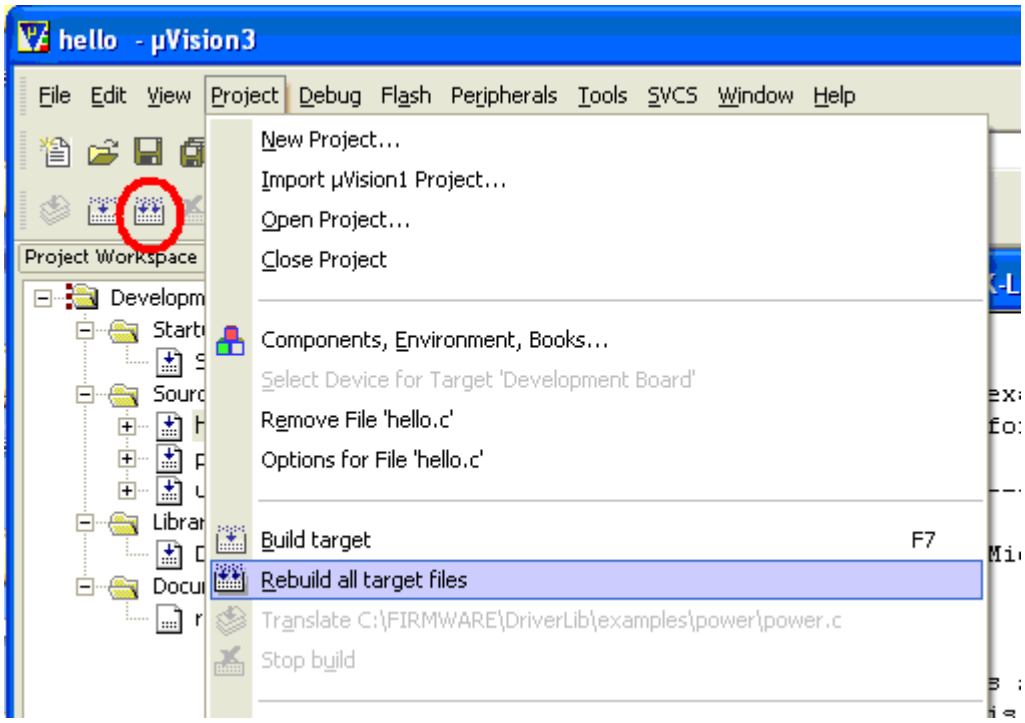


5. You can view source files in the project by double-clicking on a filename in the Project Workspace pane on the left. For example, double-click on hello.c, and the source file opens in the editor.

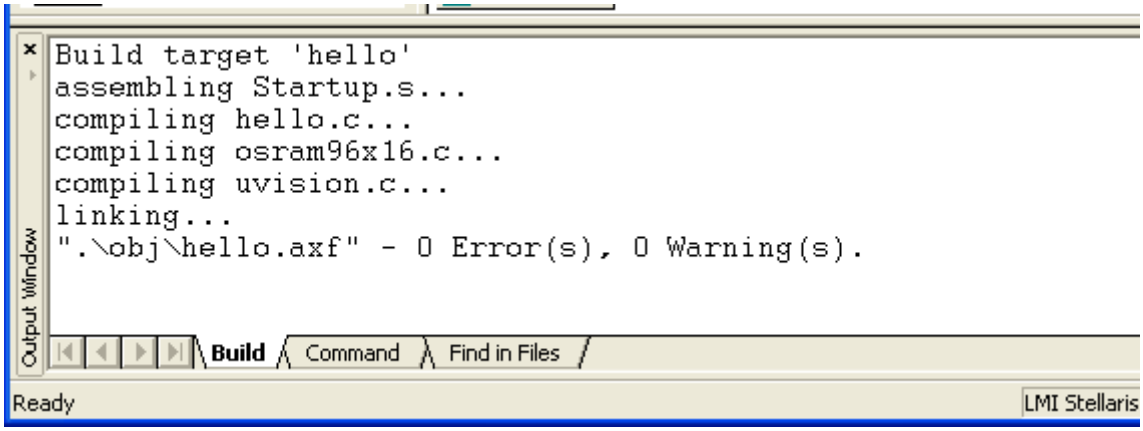


Step 4: Build the Hello Project

1. Select “Rebuild all target files” from the Project menu, or click on the “Rebuild all” button (icon).



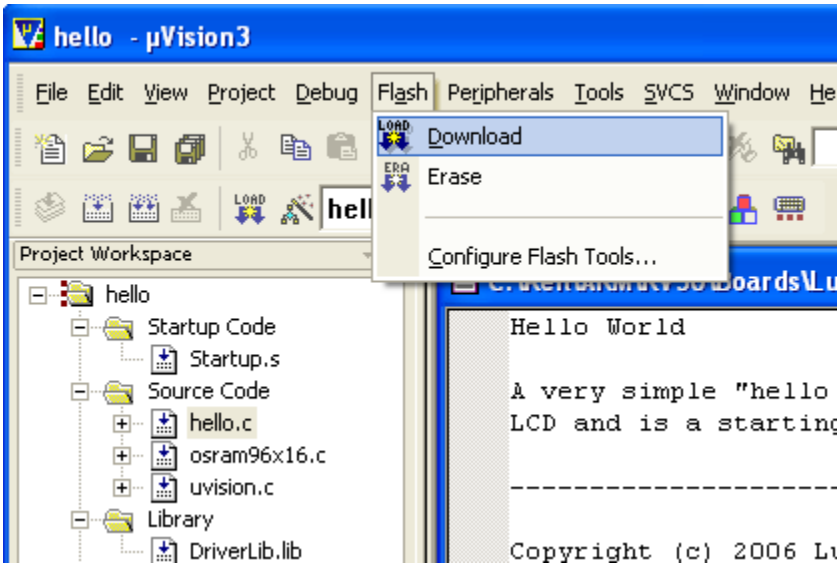
2. All of the source files are compiled and linked. The activity can be seen in the Build window at the bottom of the µVision IDE. The process completes with an application named hello.axf built with no errors and no warnings.



Step 5: Load the Hello Program into the Flash Memory

Important: The FTDI driver interface in the kit is slower than a standard ULINK interface, and is intended to be used for tool evaluation. Upgrading to a ULINK for your custom design allows for much faster download speeds. The ULINK *cannot* be used with the EKK-LM3S811, but is compatible with all other evaluation boards.

1. Select "Download" from the Flash menu, or click on the "Download" button (icon).

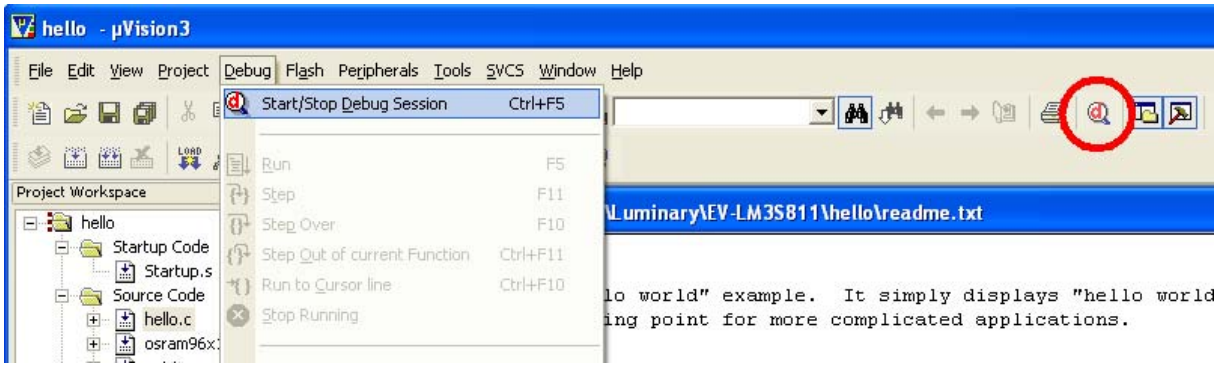


2. The process takes a few seconds. You see a progress bar at the bottom of the IDE window as the device is programmed. When it is finished, you should see in the Build window that the device was erased, programmed, and verified OK.

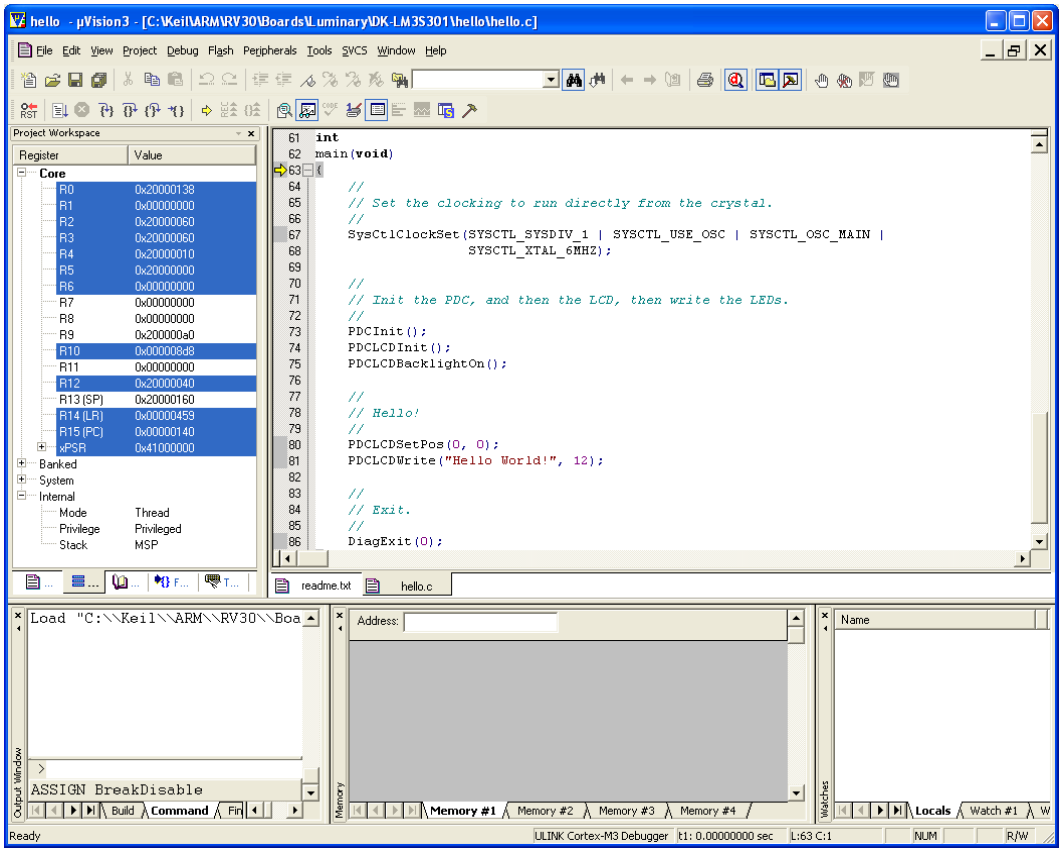
3. The Hello application is now programmed into the flash memory of the Stellaris microcontroller on the Evaluation Board.

Step 6: Debug and Run the Hello Program

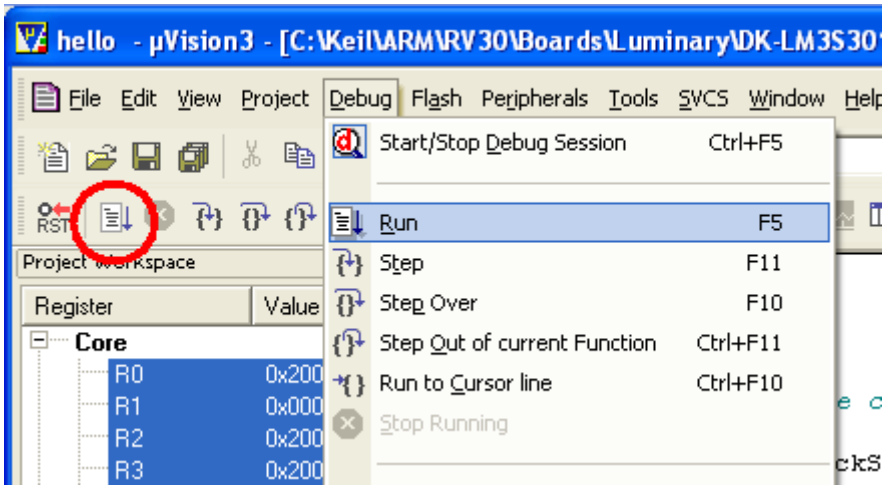
1. Select “Start/Stop Debug Session” from the Debug menu or click the “Debug” button (icon).



2. The IDE switches to debugging mode. The processor registers show up in a window on the left, the debugger command window is visible at the bottom, and the main window shows the source code being debugged. The debugger automatically stops at main.



3. From here, you can examine and modify memory, program variables and processor registers, set breakpoints, single step, and all other typical debugging activities. To run the program, select "Run" from the Debug menu, or click on the "Run" button (icon).



4. The application starts running, and you should see the text "Hello World!" output to the OLED display of the evaluation board.

Step 7: Build and Run Additional Example Programs

There are several additional example project folders under the folder:

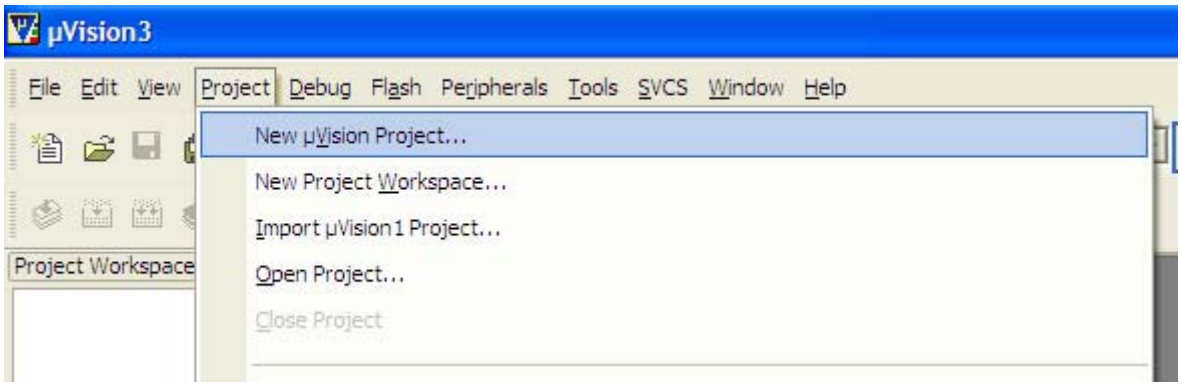
```
StellarisWare\boards\{board_name}
```

These additional example programs are discussed in the *StellarisWare Peripheral Driver Library User's Guide* (found on the Evaluation Kit Documentation and Software CD). You can open, build, and run any of these in the same way by going back to Step 4 and opening a different project, as long as it fits within the 32 KB code limit of the evaluation tools. All projects larger than 32 KB have binaries available that can be downloaded with the LMFLASH utility.

Creating a New Project

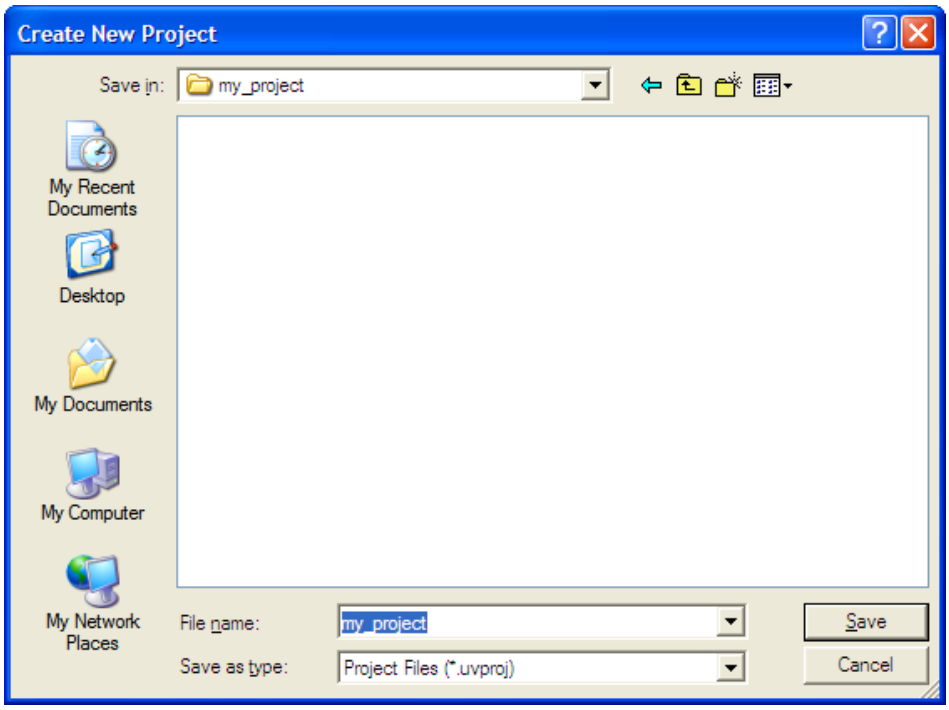
Once you have gone through the StellarisWare example applications, you may want to create your own project to start development. While you can always start with an existing, simple project, sometimes you may want to start fresh.

To create a new project, in project menu, select Project > New uVision Project…



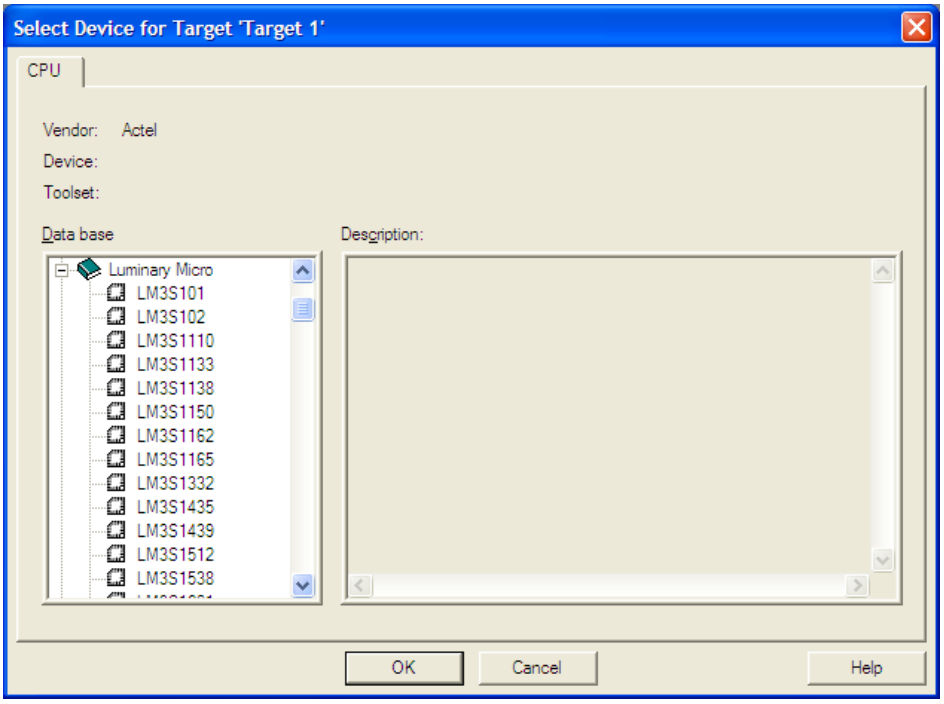
You then get a dialog box asking you where you would like to save your new project. While you can put the new project anywhere, if you plan to use the StellarisWare framework, you can create the project within the StellarisWare tree. You can put it in either the existing StellarisWare\boards directory, or create a new item in the boards directory that corresponds to either your specific board or development. In this case, we'll create a new directory called "my_board" in StellarisWare boards and create a new project called "my_project." The StellarisWare tree now looks like:

```
C:\StellarisWare\boards\my_board\my_project
```

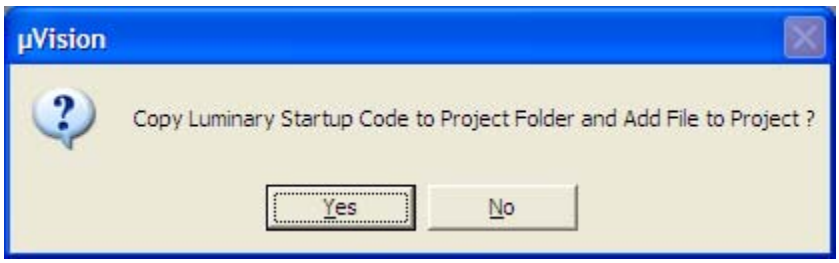


Once the project file (.uvproj) is saved, a dialog window pops up asking you to select the device that you are using. Select the appropriate device under the Luminary Micro or Texas Instruments list.

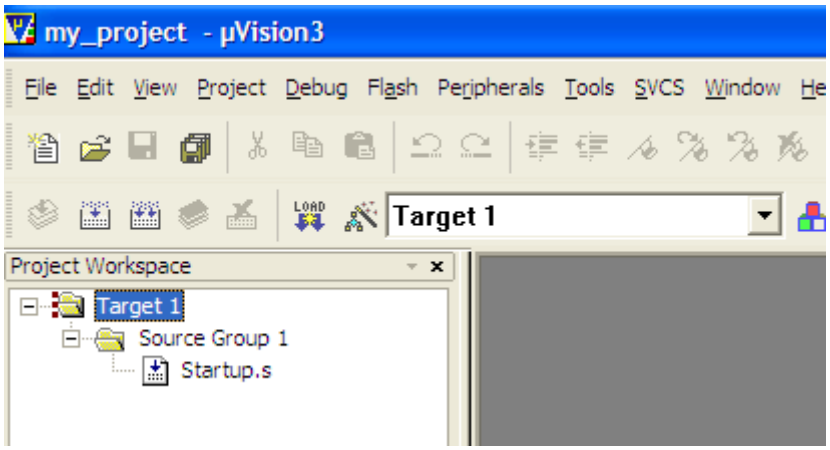
Important: A common question is "what if my device isn't in the list?" That's OK, don't worry. The device selection is mainly used to configure the memory sizes (linker). If you select a similar device to the one you have, it will work just fine.



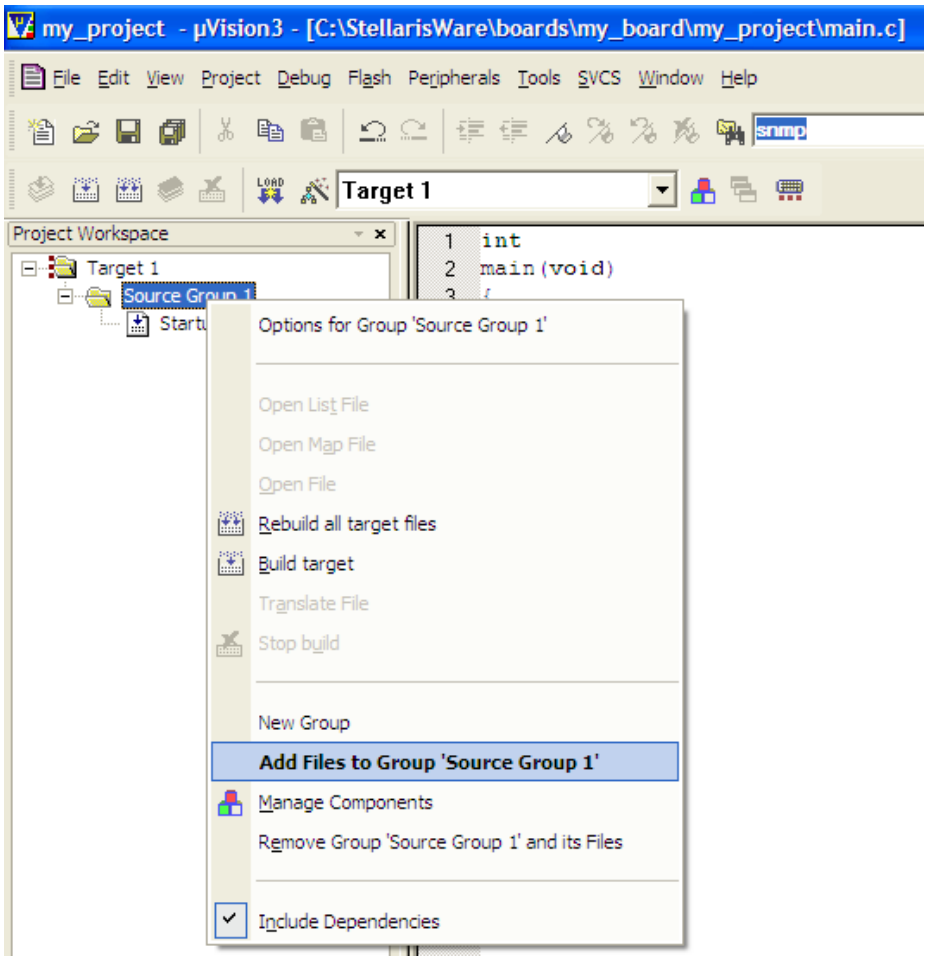
The last thing that the tool asks is whether you want to add startup code to the project. You can click yes or no, but just know that you do need to have startup code whether it's the one provided by Keil or one that you get from somewhere else. For this example, we'll add the startup code provided by the tool.



Your empty project will look like this:



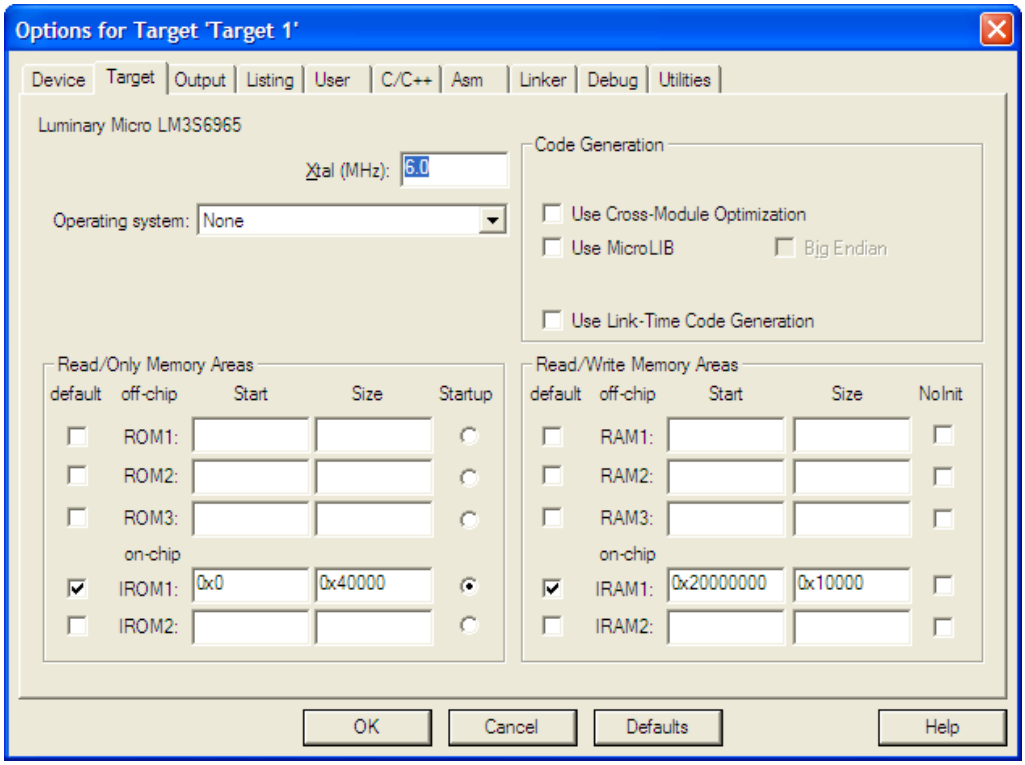
The startup code adds the vector table for you, so all you need to do is create a C file with a main function. For this example, create a new file (from File > New…) and add a simple main function with a while(1) loop; then save it as a *.c file in your project directory. Once you have saved it, right click on the "Source Group 1" folder and select "Add Files to Group 'Source Group 1'." When the dialog box pops up to find the file, browse to your project directory and select the main.c file you just created.



This process gives you the most basic project that will run on a Stellaris microcontroller. The next steps include configuring your project for your specific board, adding the hooks for StellarisWare, and setting up the debug and flash programming environment.

Configuring for your hardware:

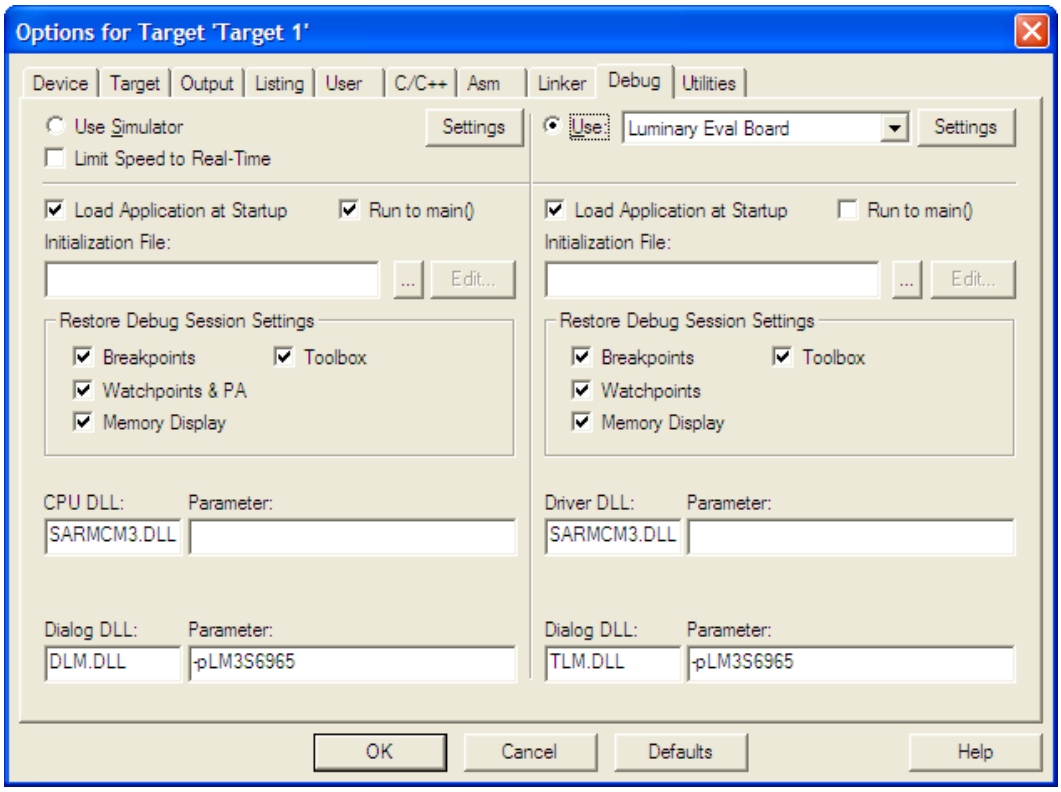
To set up the project for your specific board, go to Project > Options for Target 'Target 1.' The following dialog box appears:



Here you need to set up the Xtal frequency for your specific board. This step is important because the value of the Xtal field is used to calculate flash programming timing later on. Also check the ROM/RAM sizes to make sure that it corresponds to the device you have.

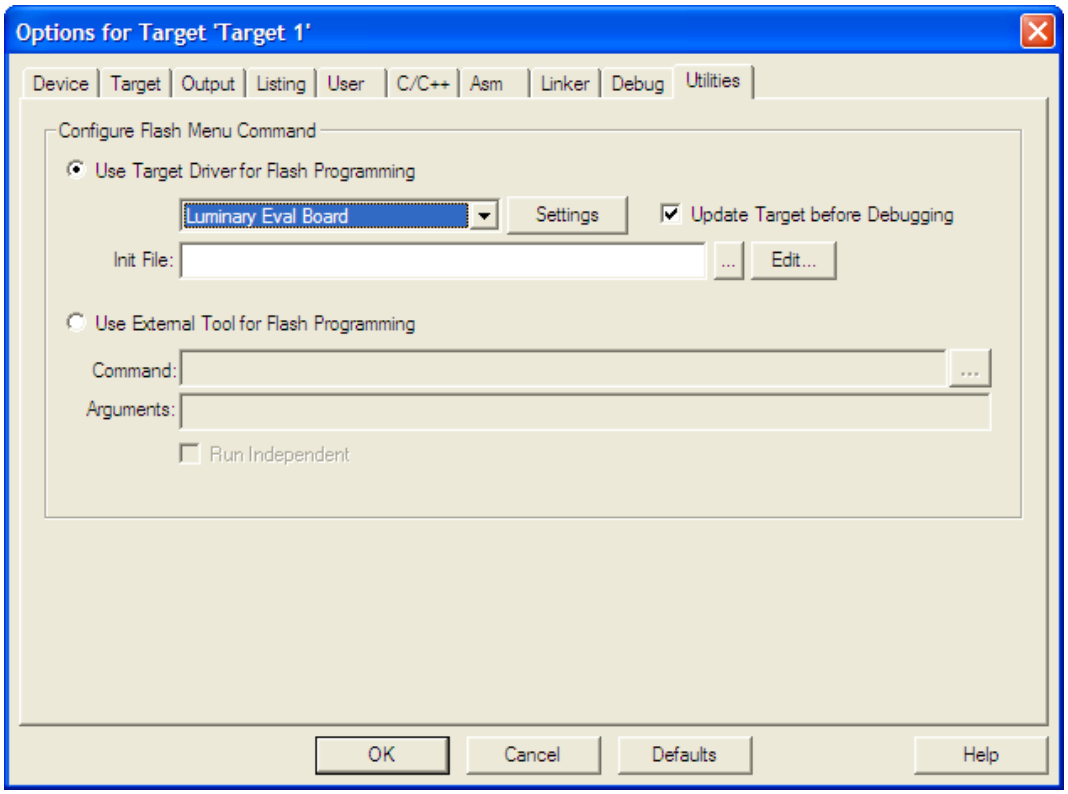
Setting up Debug:

In the same project options dialog shown above, select the Debug tab. The default configuration for a new project is to use the simulator, which is not going to allow you to debug real hardware. Choose the "Use" radio on the right side with the "ULINK Cortex Debugger" next to it (this selection is the default in the drop-down box). If you're using a Stellaris evaluation or development board as your in-circuit debug interface (ICDI), choose the "Luminary Eval Board" option from the drop-down box.



Setting up Flash Programming:

Just as you did for debug, you must select the appropriate debug interface for flash programming. Browse to the Utilities tab of the project options and select the "Luminary Eval Board" option from the drop-down box.



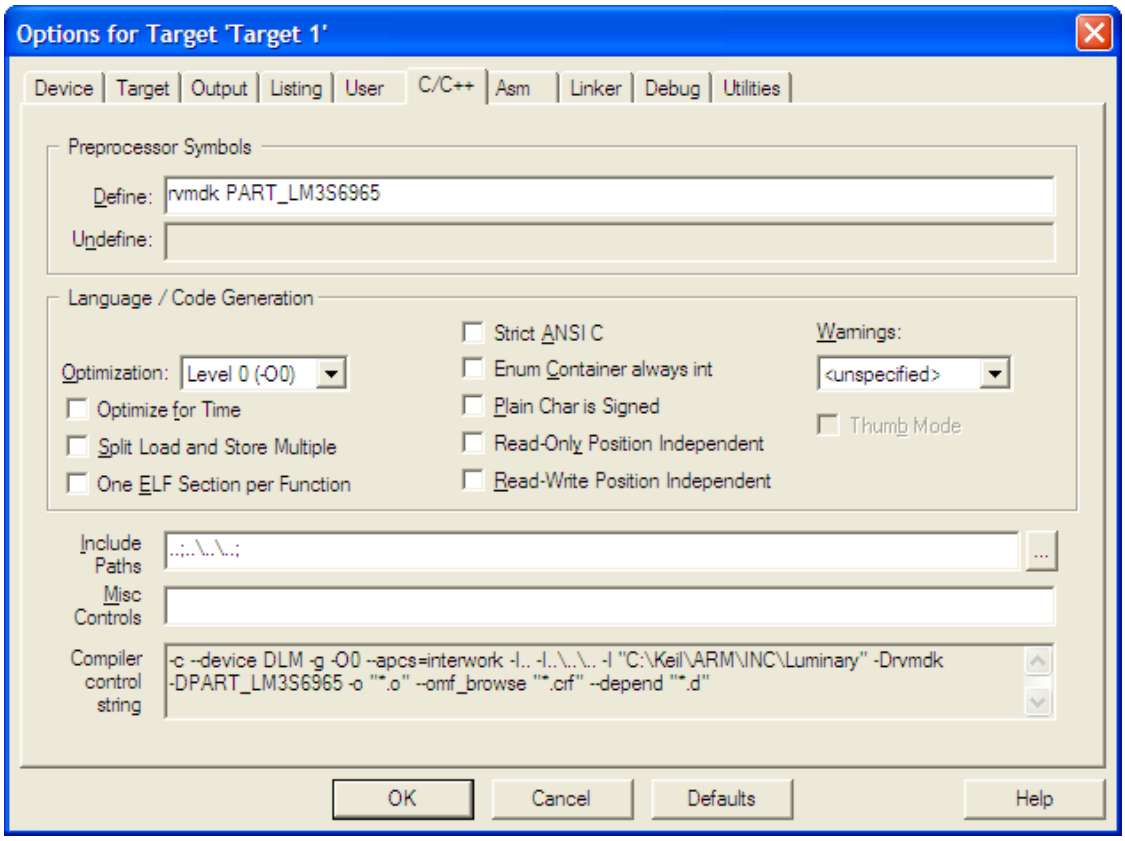
With these settings complete, you will be able to download and debug your simple application.

Adding StellarisWare hooks:

The easiest way to see how to configure your project to use the StellarisWare drivers and utilities is to look at the examples. The basic things you'll need to do are:

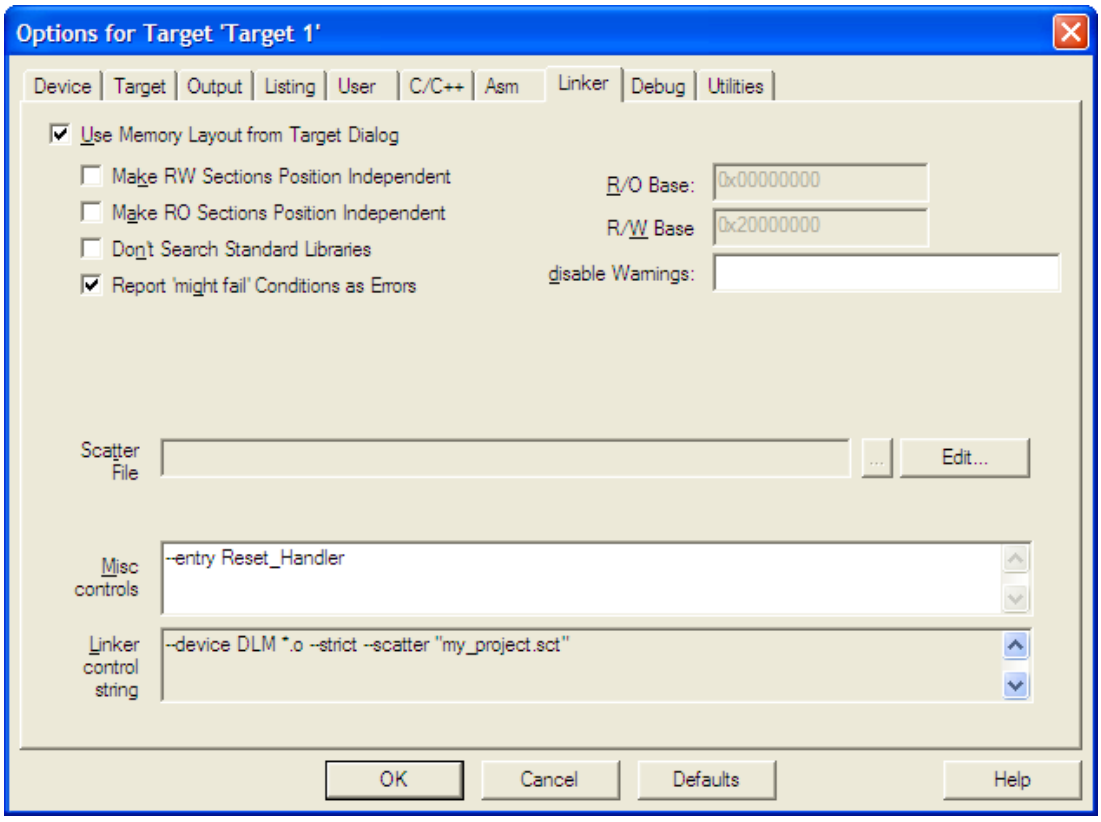
1. Add the driverlib.lib file to your project. This gives the application the entire driver source at compile/link time. To actually use any of the functions, you'll need to include the appropriate header files. To add driverlib.lib to your project, right click on the "Source Group 1" folder and select "Add Files to Group 'Source Group 1'." Browse to StellarisWare\driverlib\rvmdk and select the driverlib.lib file. Note, you need to tell the file browser to look for *.lib file types, so change the "Files of type" drop-down from "C Source file (*.c)" to "Library file (*.lib)" or "All files".

2. In your project options, select the C/C++ tab. Here you need to tell the project that you're using the ARM compiler, so you should define "rvmdk." This definition is used in StellarisWare to compile the Keil specific sections correctly. You should also define the part you're using as well as add the top level StellarisWare directory to your include path.



3. In the linker section, you must tell the application where the entry point is, so add "—entry Reset_Handler" to the Miscellaneous controls box. If your startup code changes, you should adjust this accordingly.

   The default linker configuration is to use an automatically generated linker/scatter file, which is fine for most people. If you want to use your own scatter file, you should uncheck the "Use Memory Layout from Target Dialog" and point the tool to a scatter file.



These steps get the basic StellarisWare hooks into your project. The best way to make sure you've done everything correctly is to go over some of the examples that we provide in the StellarisWare package.

Conclusion

You have now installed the Keil RealView Microcontroller Development Kit and used it to build, load, and run a demonstration application on your Stellaris Evaluation Board. You have also learned how to create a new project from scratch. From here, you can experiment with the debugger or start creating your own application from the ground up or using the Hello program as an example.

References

The following references are included on the Stellaris Evaluation Kit Documentation and Software CD and are also available for download at www.ti.com/stellaris:

- *Stellaris Evaluation Kit User's Manual*
- StellarisWare Software, Order Number SW-LM3S
- *StellarisWare Peripheral Driver Library User's Guide,* Order Number SW-DRL-UG

In addition, the following website may be useful:

- RealView MDK website at http://www.keil.com/arm/rvmdkkit.asp

Texas Instruments
108 Wild Basin Rd., Suite 350
Austin, TX 78746
http://www.ti.com/stellaris

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265