Keil µVision for ModusToolbox™ user guide

ModusToolbox™ tools package version 3.1.0

About this document

Scope and purpose

ModusToolbox™ software is a set of tools and libraries that support device configuration and application development. These tools enable you to integrate our devices into your existing development methodology. This document provides information and instructions for using Keil µVision with ModusToolbox™ software.

Document conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, menus and sub-menus.
Italics	Denotes file names and paths.
`Courier New`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets.
File > New	Indicates that a cascading sub-menu opens when you select a menu item.

Reference documents

Refer to the following documents for more information as needed:

- ModusToolbox™ tools package installation guide –Provides information and instructions about installing the tools package on Windows, Linux, and macOS.
- ModusToolbox™ tools package user guide –Provides information about all the tools included with ModusToolbox™ tools package.
- Dashboard user guide –Provides specific information about the Dashboard tool.
- Project Creator user guide –Provides specific information about the Project Creator tool.
- Device Configurator guide –Provides specific information about the Device Configurator.

Table of contents

1 Download/install software

1.1 ModusToolbox™ tools package

Refer to the instructions in the ModusToolbox™ tools package installation guide for how to download and install the ModusToolbox™ tools package.

1.2 Keil µVision (Windows only)

Keil µVision version 5.30 or later

1.3 Python

Python 3.8 is installed in the tools_3.x directory, and the make build system has been configured to use it. You don't need to do anything if you use the modus-shell/Cygwin.bat file to run command line tools.

However, if you plan to use your own version of Cygwin or some other type of bash, you will need to ensure your system is configured correctly to use Python 3.8. Use the `CY_PYTHON_PATH` as appropriate.

1.4 J-Link

For J-Link debugging, download and install J-Link software:

https://www.segger.com/downloads/J-Link/J-Link_Windows.exe

2 Getting Started

This section covers the ways to get started using IAR Embedded Workbench with ModusToolbox™ software.

- Create new application
- Exporting existing application
- Open application Keil µVision

2.1 Create new application

Creating an application includes several steps, as follows:

2.1.1 Step 1: Open Project Creator tool

The ModusToolbox™ Project Creator tool is used to create applications based on code examples and template applications. You can open the Project Creator tool from the Windows **Start** menu, or by launching the executable. By default, the tool is installed in the following directory:

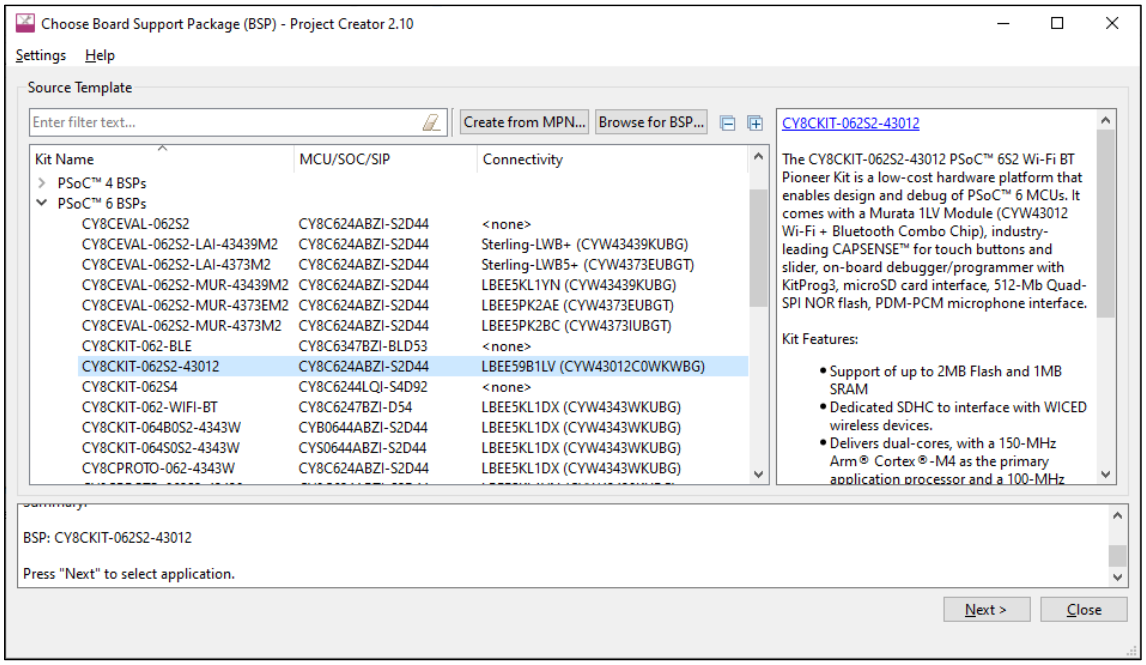
<user_home>/ModusToolbox/tools_<version>/project-creator

The tool is provided in GUI form and as a command line interface. For more details, refer to the Project Creator user guide.

Note: You can also launch the Project Creator tool from the ModusToolbox™ Dashboard. Refer to the Dashboard user guide for details.

2.1.2 Step 2: Choose Board Support Package (BSP)

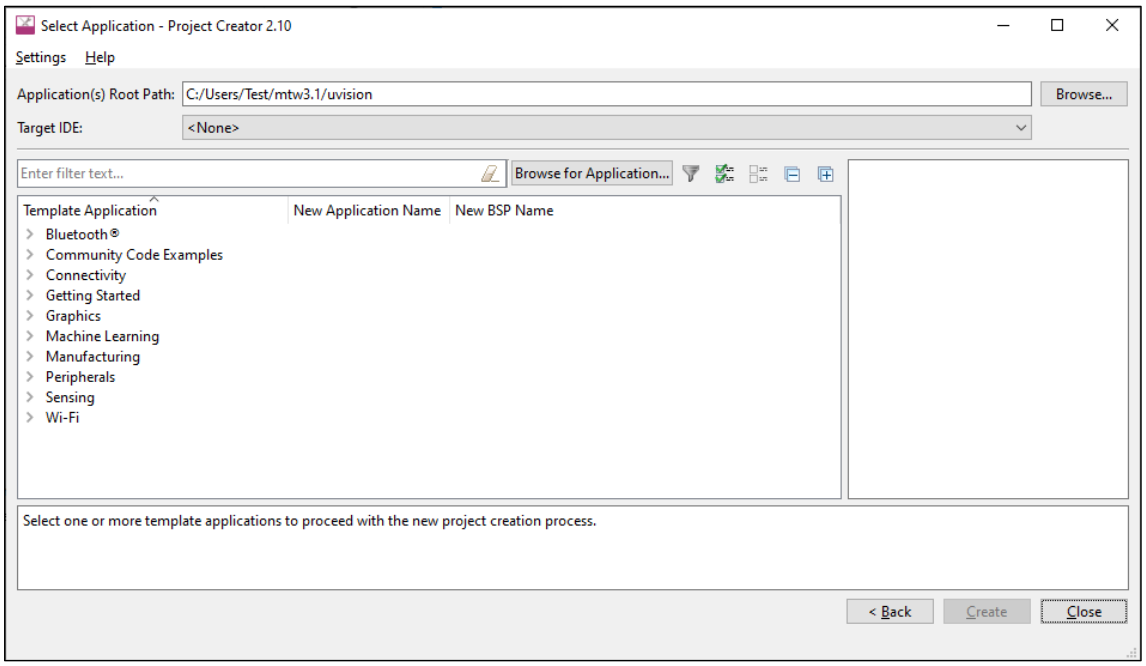
When the Project Creator tool opens, expand one of the BSP categories under **Kit Name** and select an appropriate kit; see the description for it on the right. The following image is an example; the precise list of boards available in this version will reflect the platforms available for development. You can also create a new BSP or browse for one on disk.



For this example, select the **CY8CKIT-062S2-43012** kit.

2.1.3 Step 3: Select application

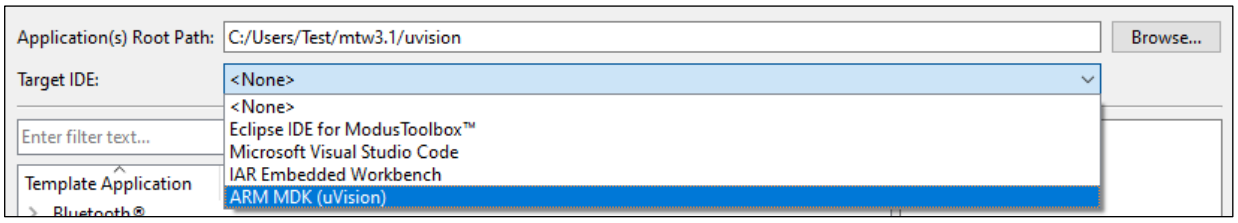
Click **Next >** to open the Select Application page.



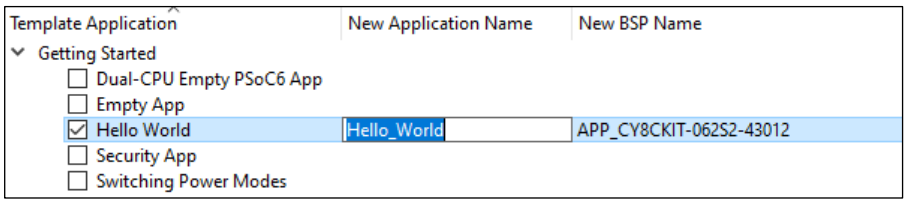
This page displays example applications, which demonstrate different features available on the selected BSP. In this case, the CY8CKIT-062S2-43012 provides the PSoC™ 62 MCU and the AIROC™ CYW43012 Wi-Fi & Bluetooth® combo chip. You can create examples for PSoC™ 6 MCU resources such as CAPSENSE™ and QSPI, as well as numerous examples for other capabilities.

Click **Browse...** next to **Application(s) Root Path** to create or specify a folder where the application will be created.

Pull down the **Target IDE** menu, and select **ARM MDK (uVision)**.



Under the **Template Application** column, expand **Getting Started** and select **Hello World** from the list. This example exercise uses the PSoC™ 6 MCU to blink an LED.



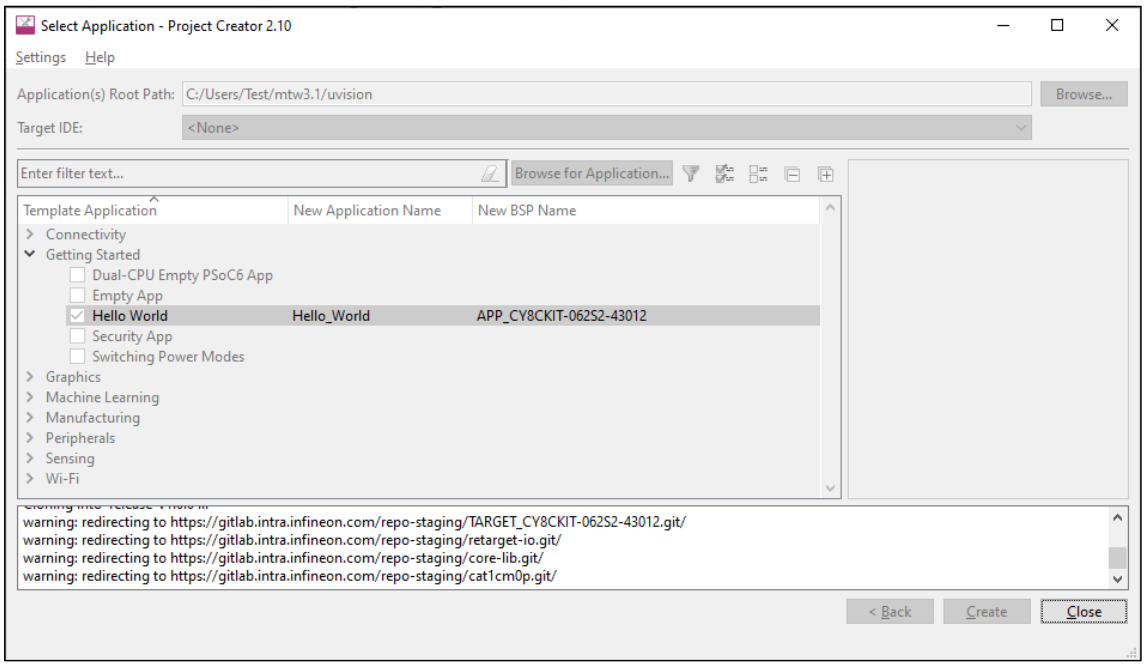
The actual application names available might vary.

Type a name for your application and/or BSP, or leave the default names. Do not use spaces. Also, do not use common illegal characters, such as:

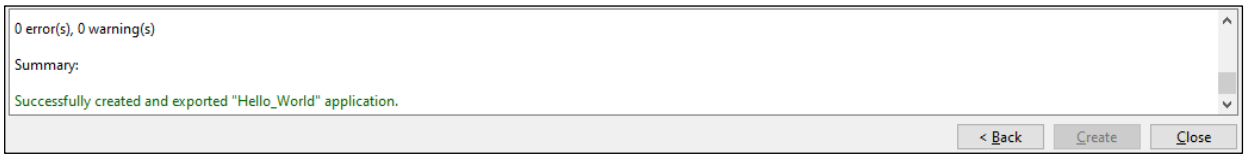
```
* . " ` / \ [ ] : ; | = ,
```

2.1.4 Step 4: Create application

Click **Create** to start creating the application. The tool displays various messages.



When the process completes, a message states that the application was created. Click **Close** to exit the Project Creator tool.



2.2 Export existing application

If you have a ModusToolbox™ application that was created for another IDE or for the command line, you can export that application to be used in µVision. Open a terminal window in the application directory, and run the command `make uvision TOOLCHAIN=ARM`.

Note: *For applications that were created using core-make-3.0 or older, you must use the* `make uvision5` *command instead.*

Note: *This sets the TOOLCHAIN to ARM in the Keil µVision configuration files but* ***not*** *in the ModusToolbox™ application's Makefile. Therefore, builds inside Keil µVision will use the ARM toolchain while builds from the ModusToolbox™ environment will continue to use the toolchain that was previously specified in the Makefile. You can edit the Makefile's TOOLCHAIN variable if you also want ModusToolbox™ builds to use the ARM toolchain.*

Note: *Check the output log for instructions and information about various flags.*

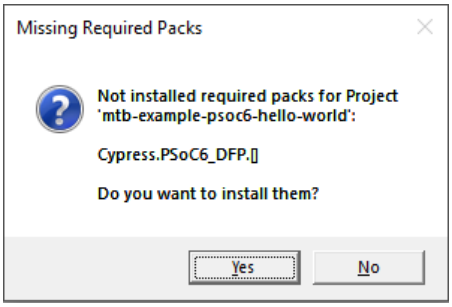
2.3 Open application in Keil µVision

Creating or exporting the application generates the following file in the application directory:

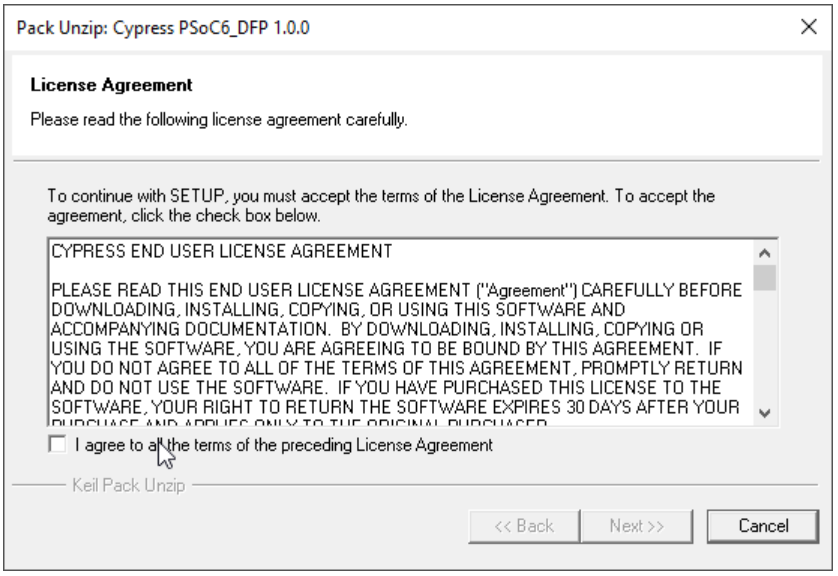
mtb-example-psoc6-hello-world.cprj

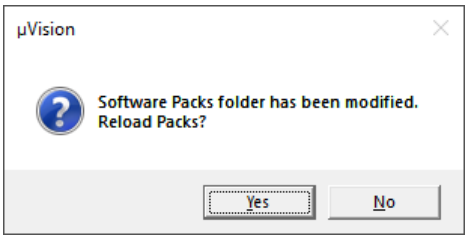
The cprj file extension should have the association enabled to open it in Keil µVision.

1. Double-click the *mtb-example-psoc6-hello-world.cprj* file. This launches the Keil µVision IDE. The first time you do this, the following dialog displays:



2. Click **Yes** to install the device pack. You only need to do this once.
3. Follow the steps in the Pack Installer to properly install the device pack.





Note: *In some cases, you may see the following error message:*

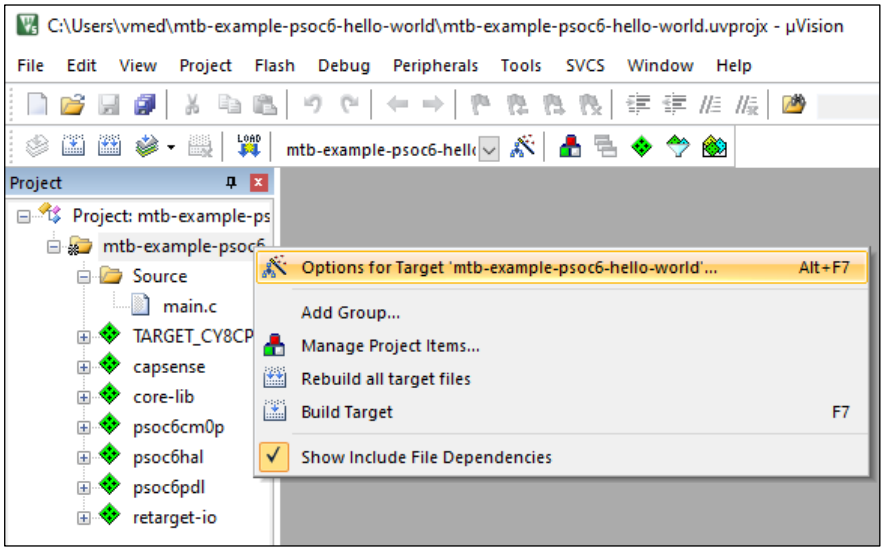
SSL caching disabled in Windows Internet settings. Switched to offline mode.

See this link for how to solve this problem:
https://developer.arm.com/documentation/ka002253/latest

4. When complete, close the Pack Installer and close the Keil µVision IDE.
5. Then double-click the *.cprj* file again and the application will be created for you in the IDE.

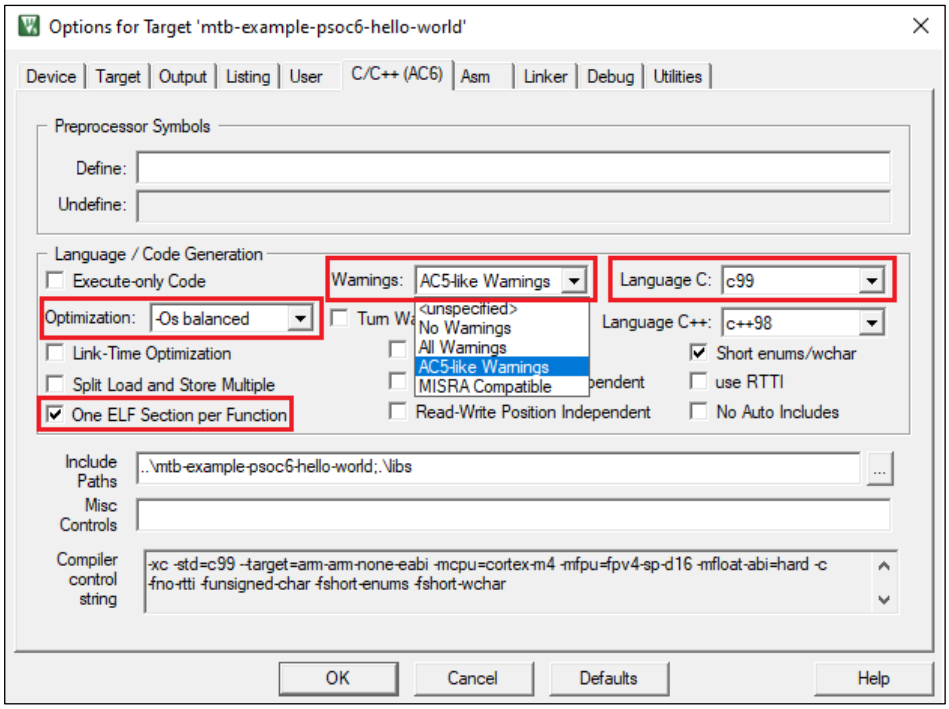
3 Configure and build the application

1. Right-click on the *mtb-example-psoc6-hello-world* directory in the µVision Project view, and select **Options for Target '<application-name>'** …



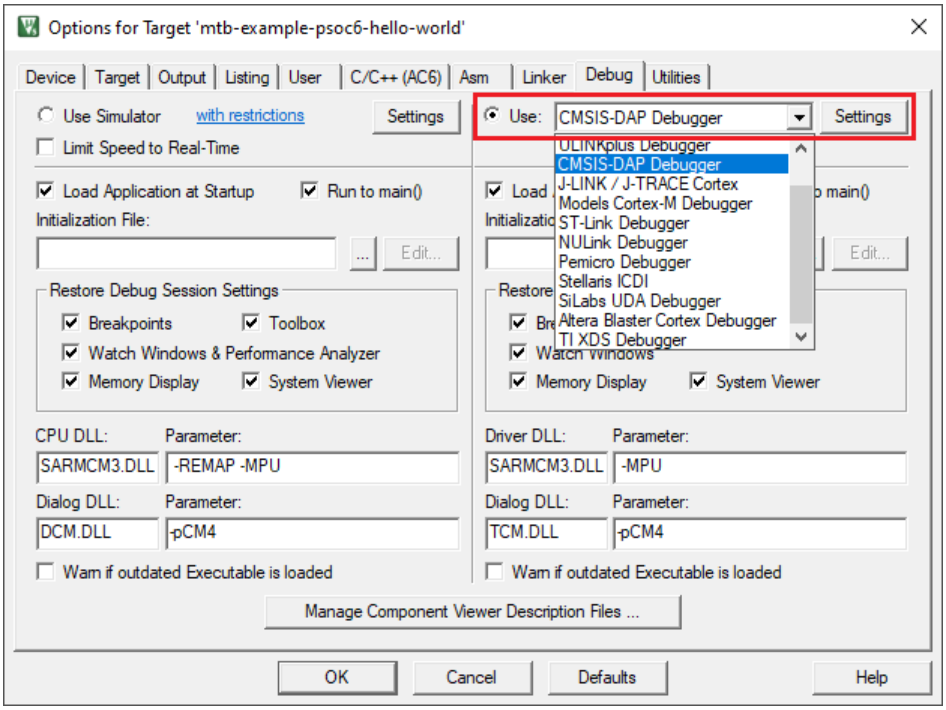
2. On the dialog, select the **C/C++ (AC6)** tab.

   - Check that the **Language C** version was automatically set to c99.
   - Select "AC5-like warnings" in the **Warnings** drop-down list.
   - Select "-Os balanced" in the **Optimization** drop-down list.
   - To reduce memory usage, select the **One ELF Section per Function** check box.



Configure and build the application

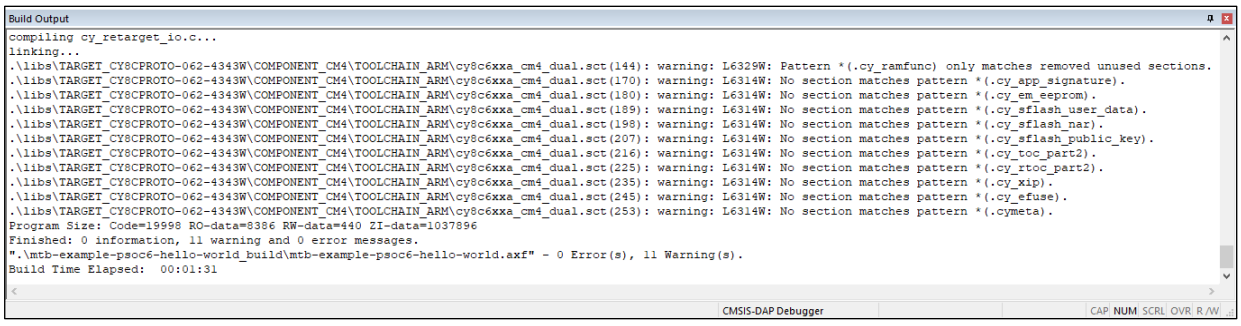
3. Select the **Debug** tab, and select KitProg3 CMSIS-DAP as an active debug adapter:



4. Click **OK** to close the Options dialog.

5. Select **Project > Build target**.

Note: *For applications using the PSoC™ 64 MCU, skip this step. Instead, perform the steps outlined in the following section.*



To suppress the linker warnings about unused sections defined in the linker scripts, add "6314,6329" to the **Disable Warnings** setting in the Project Linker Options.

3.1 PSoC™ 64 application configuration

Before building an application for a PSoC™ 64 secure MCU in Keil µVision, you must perform the following configuration steps:

1. Build the application using the ModusToolbox™ make build command. You can do this by using a Terminal or by exporting the application to Eclipse or VS Code.

Configure and build the application

2. Copy the post-build command from the log. For example:

```
C:/Infineon/Tools/ModusToolbox/tools_3.1/python/python.exe C:/UG/CY8CPROTO-064S1-SB_Secure_Blinky_LED_FreeRTOS_UG/bsps/TARGET_APP_CY8CPROTO-064S1-SB/psoc64_postbuild.py --core CM4 --secure-boot-stage single --policy policy_single_CM0_CM4 --target cyb06xx7 --toolchain-path C:/Infineon/Tools/ModusToolbox/tools_3.1/gcc --toolchain GCC_ARM --build-dir C:/UG/CY8CPROTO-064S1-SB_Secure_Blinky_LED_FreeRTOS_UG/build/APP_CY8CPROTO-064S1-SB/Debug --app-name mtb-example-psoc6-secure-blinkyled-freertos --cm0-app-path ../mtb_shared/cat1cm0p/release-v1.0.0/COMPONENT_CAT1A/COMPONENT_CM0P_SECURE --cm0-app-name psoc6_01_cm0p_secure
```

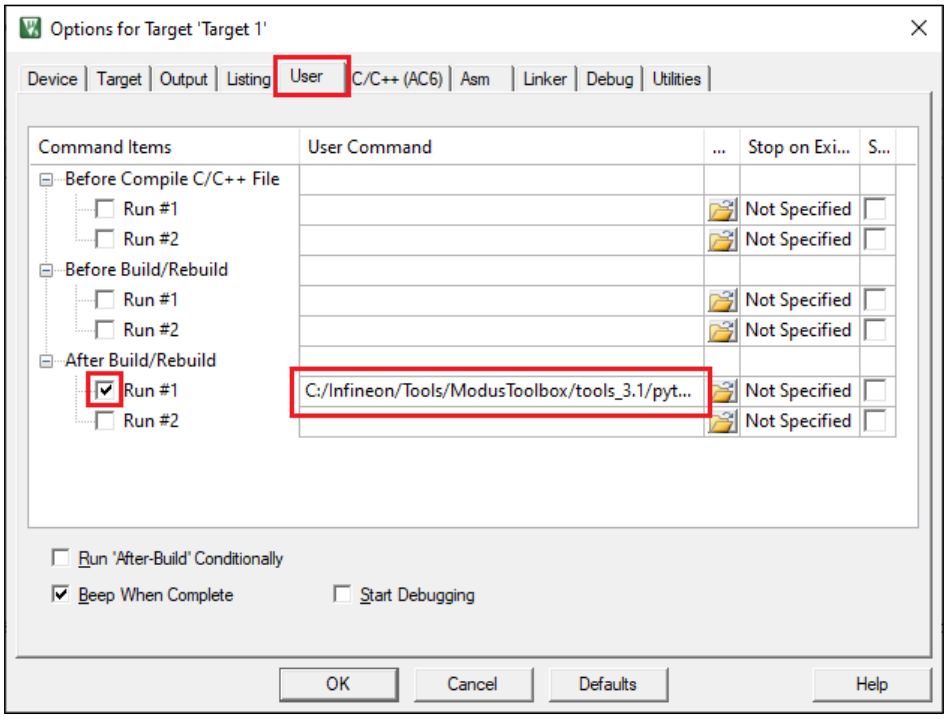
3. Paste the command into an appropriate editor, and make the following edits:

   - Change `--toolchain-path` to the ARM toolchain; for example, *C:/Keil_v5/ARM/ARMCLANG*
   - Change `--toolchain` to ARM
   - Change `--build-dir` to Keil µVision build directory

   Example of command after edit:

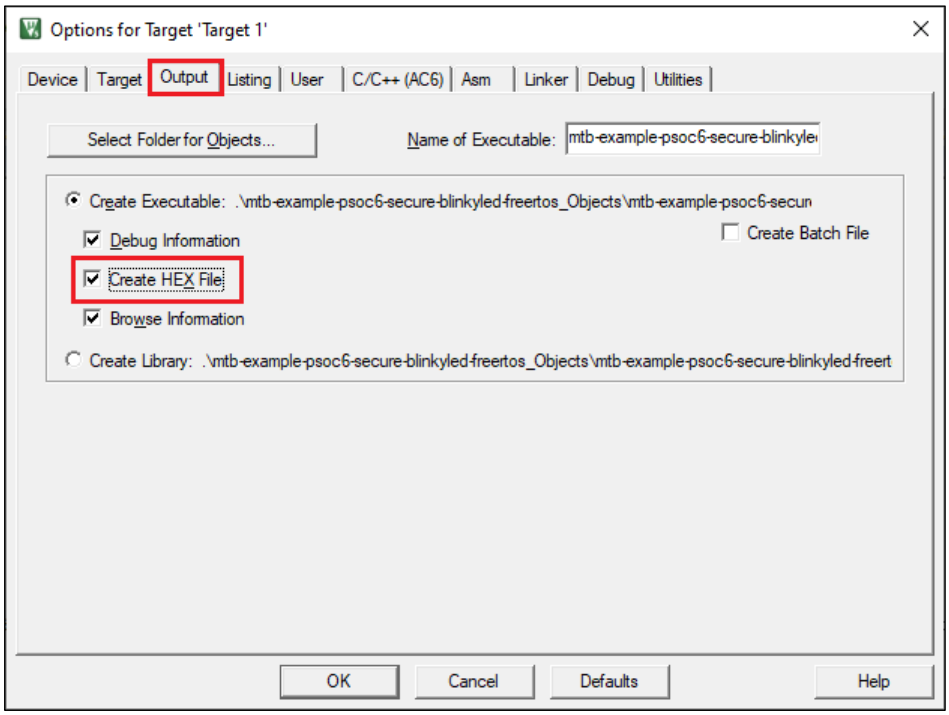
```
C:/Infineon/Tools/ModusToolbox/tools_3.1/python/python.exe C:/UG/CY8CPROTO-064S1-SB_Secure_Blinky_LED_FreeRTOS_UG/bsps/TARGET_APP_CY8CPROTO-064S1-SB/psoc64_postbuild.py --core CM4 --secure-boot-stage single --policy policy_single_CM0_CM4 --target cyb06xx7 --toolchain-path C:/Keil_v5/ARM/ARMCLANG --toolchain ARM --build-dir C:/UG/CY8CPROTO-064S1-SB_Secure_Blinky_LED_FreeRTOS_UG/mtb-example-psoc6-secure-blinkyled-freertos_Objects --app-name mtb-example-psoc6-secure-blinkyled-freertos --cm0-app-path ../mtb_shared/cat1cm0p/release-v1.0.0/COMPONENT_CAT1A/COMPONENT_CM0P_SECURE --cm0-app-name psoc6_01_cm0p_secure
```

4. Copy the edited command.
5. In µVision, select **Project > Options for Target 'Target 1'...** to open the Options dialog.
6. Select the **User** tab and enable the **Run #1** check box under **After Build/Rebuild**. Then, paste the edited command.

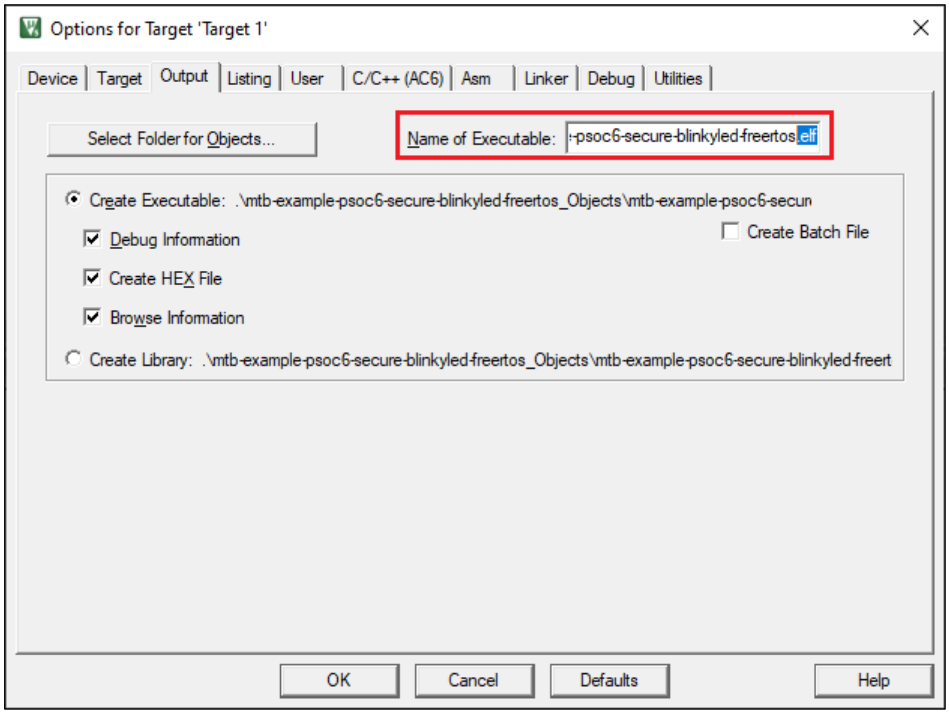


Configure and build the application

7. Select the **Output** tab and enable the **Create HEX File** check box.



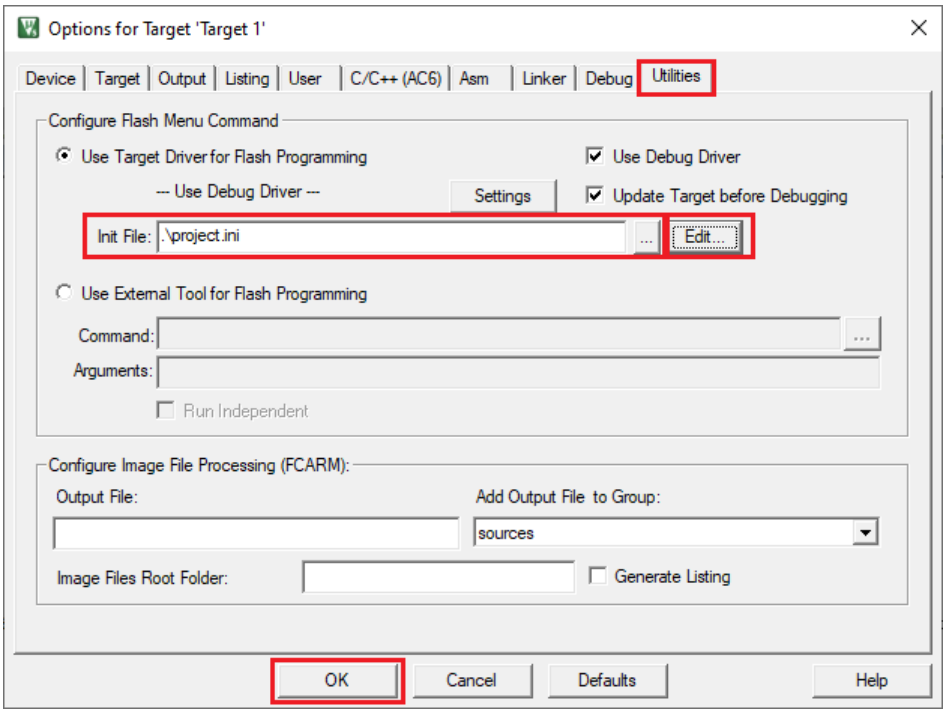
8. On the same **Output** tab, add an ".elf" extension in the **Name of Executable** field.



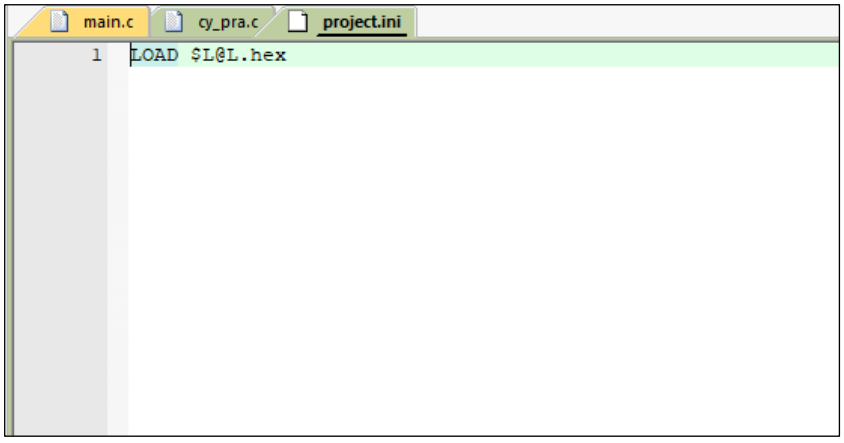
9. Create an empty *project.ini* file in the project root folder.

Configure and build the application

10. On the Options dialog, select the **Utilities** tab and select the *project.ini* file using the **[...]** button. Then, click **Edit** to open the file and click **OK** to close the options dialog.



11. Type `LOAD $L@L.hex` in the *project.ini* file and save the file.



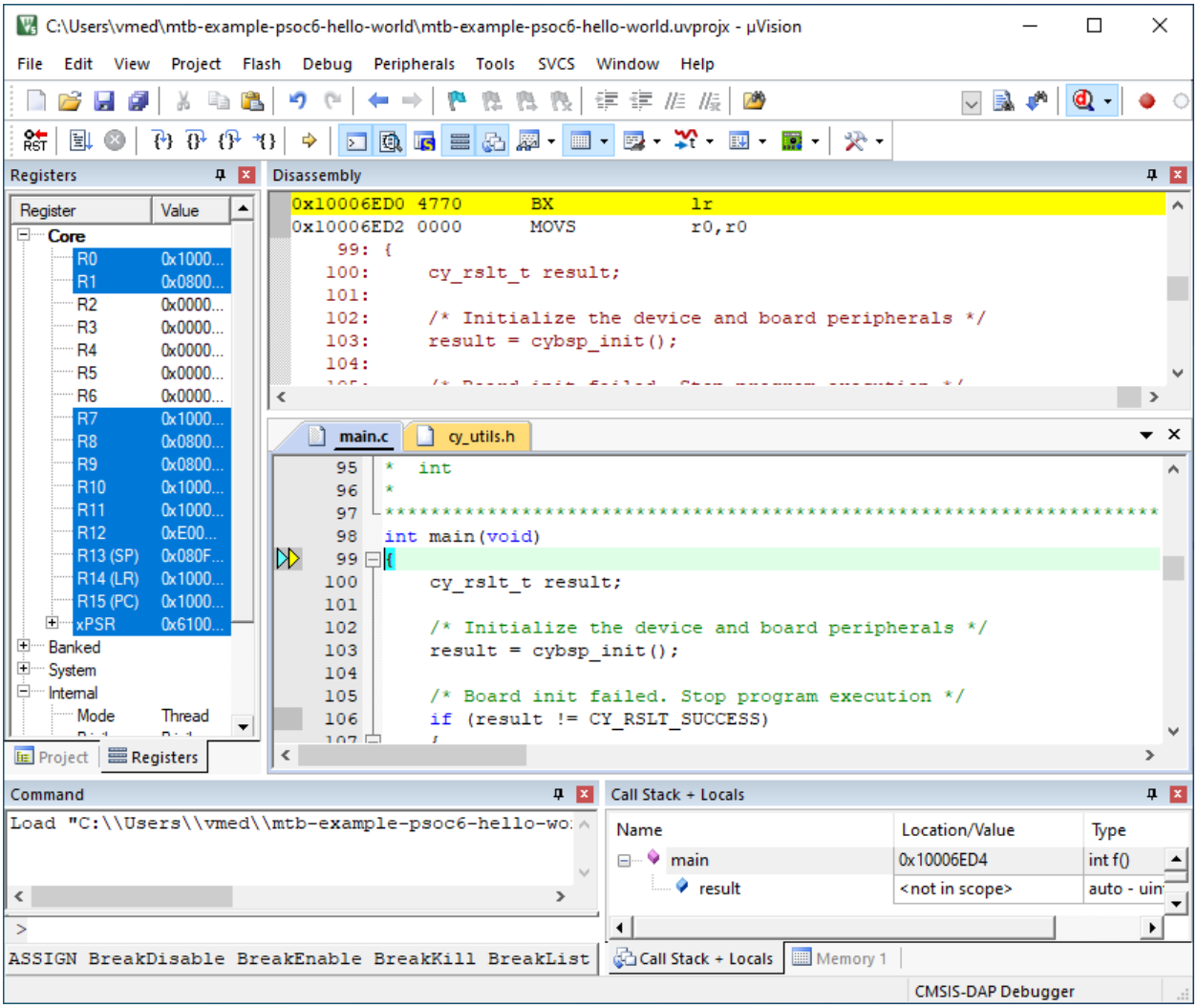
12. Select **Project > Build Target** to build the application and execute post-build commands.

After performing these steps, you should be able to run debug, erase, and program for PSoC™ 64 secure MCUs.

4 Programming/Debugging

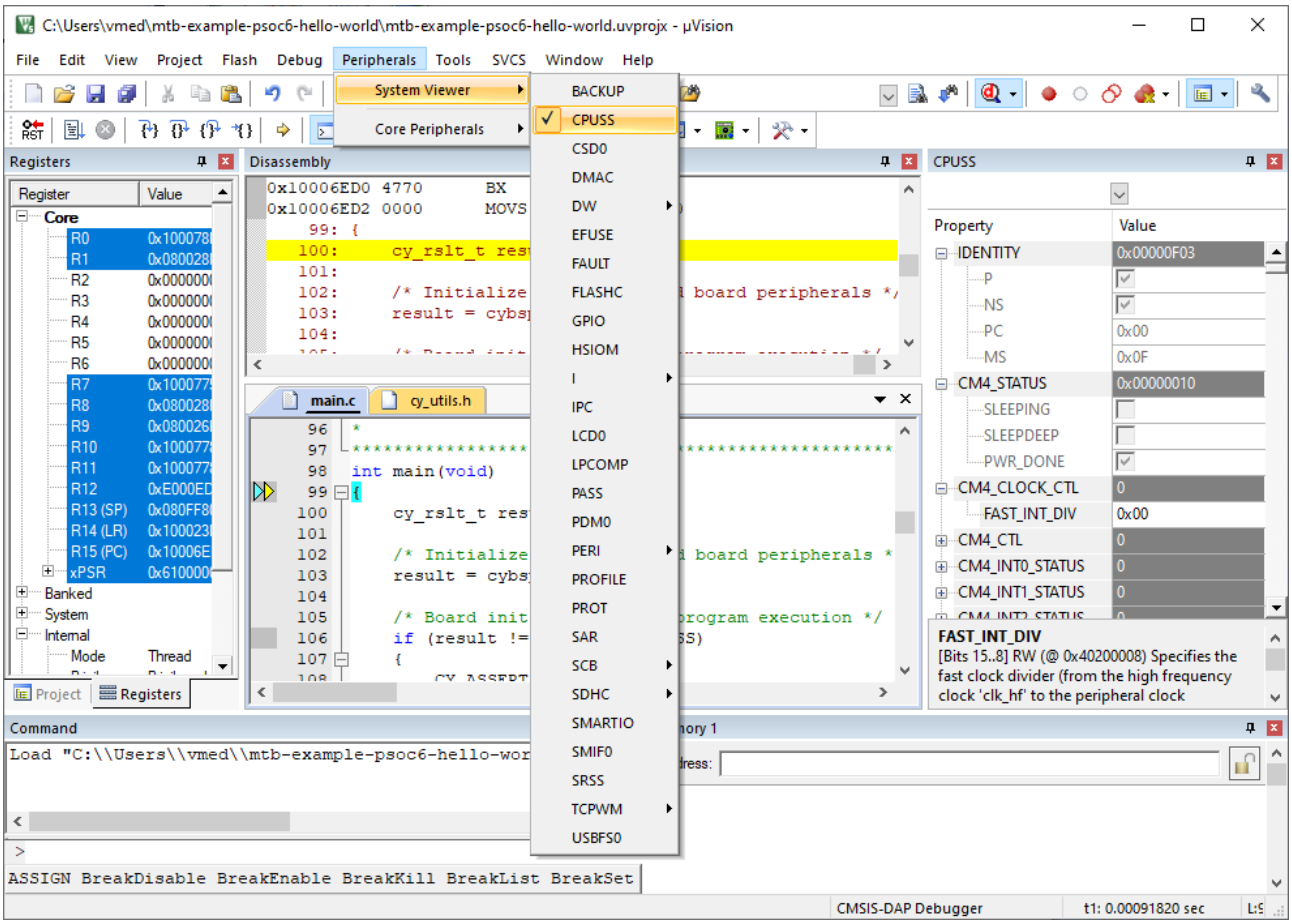
1. Connect the PSoC™ 6 kit to the host PC.
2. As needed, run the fw-loader tool to make sure the board firmware is upgraded to KitProg3. See KitProg3 User Guide for details. The tool is located in this directory by default:

   <user_home>/ModusToolbox/tools_3.1/fw-loader/bin/
3. Select **Debug > Start/Stop Debug Session.**



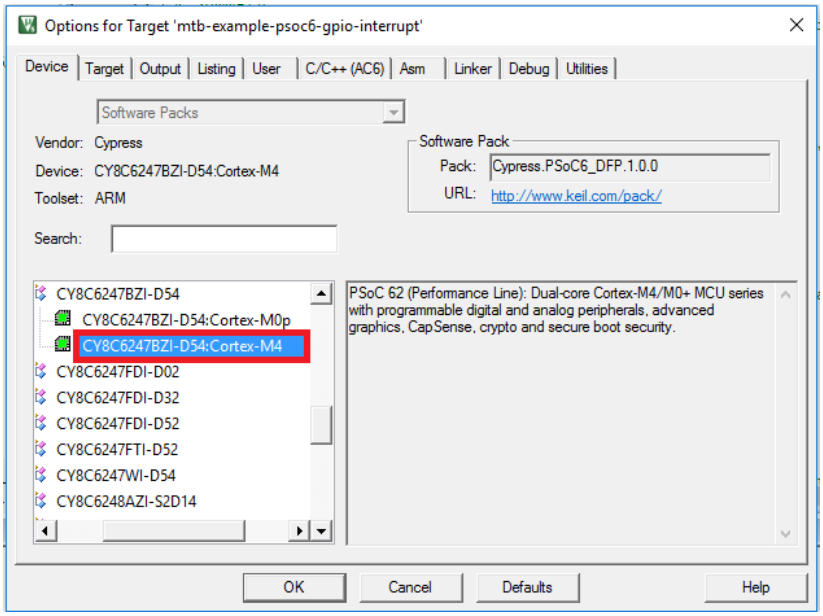
Programming/Debugging

You can view the system and peripheral registers in the SVD view.



4.1 To use KitProg3/MiniProg4, CMSIS-DAP, and ULink2 debuggers

1. Select the **Device** tab in the Options for Target dialog and check that M4 core is selected:

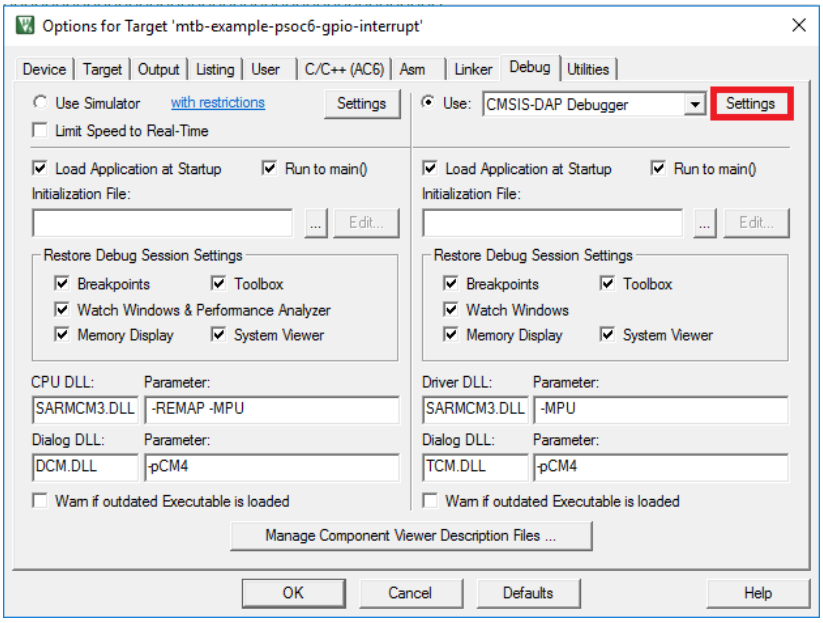


Programming/Debugging

2. Select the **Debug** tab.

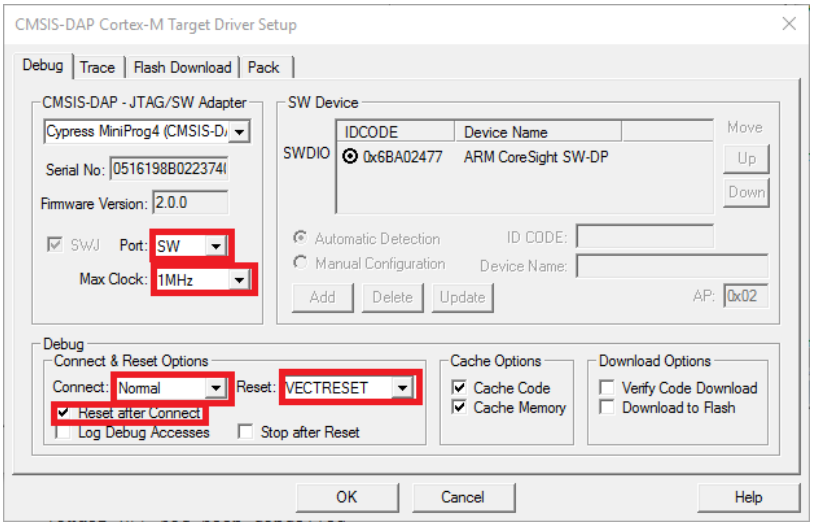
Note: *To use the ULink2 probe for multi-core debugging, select the **CMSIS-DAP Debugger** instead of ULink for each core of the project (CM4 and CM0P).*

3. Click the **Settings** button.



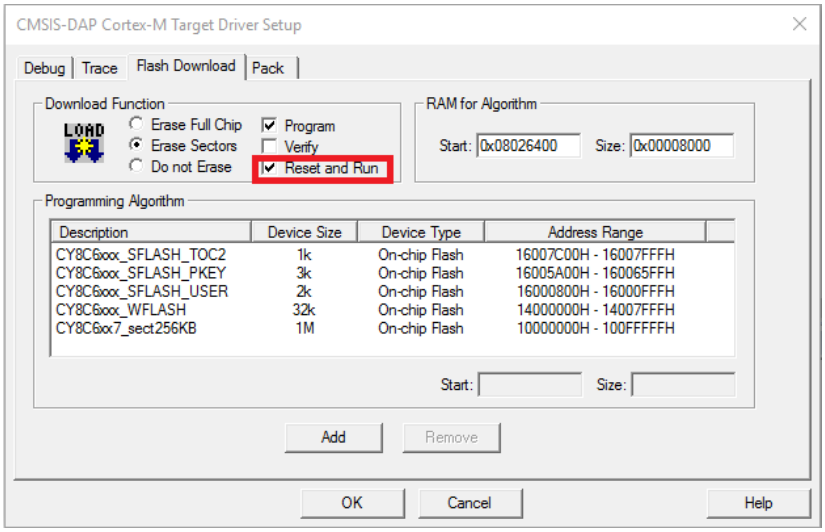
4. On the Target Driver Setup dialog on the **Debug** tab, select the following:

- set **Port** to "SW"
- set **Max Clock** to "1 MHz"
- set **Connect** to "Normal"
- set **Reset**:
  - For PSoC™ 6, to "VECTRESET"
  - For PSoC™ 4, PMG1, and AIROC™ CYW208xx, to "SYSRESETREQ"
- enable **Reset after Connect** option

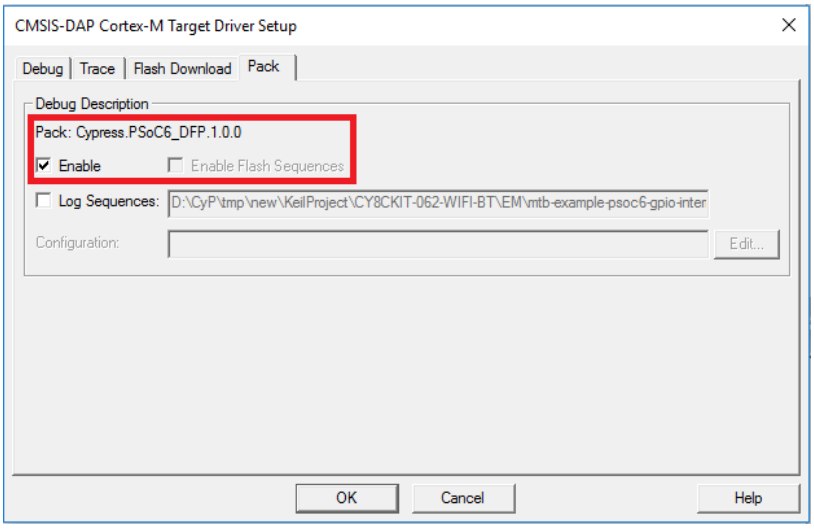


Programming/Debugging

5. Select the **Flash Download** tab and select "Reset and Run" option after download, if needed:

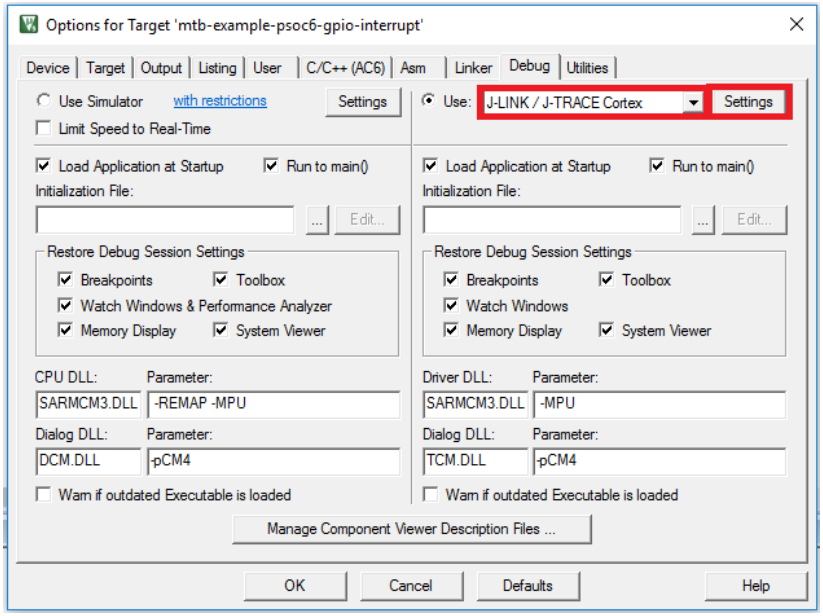


6. Select the **Pack** tab and check if "Cypress.PSoC6_DFP" is enabled:

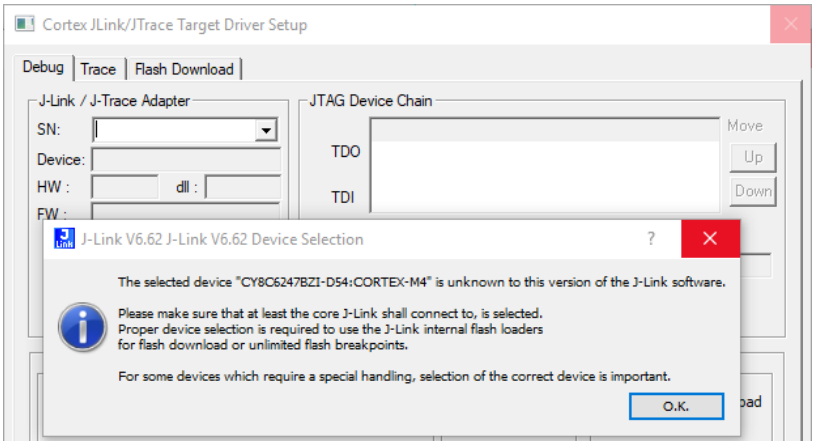


4.2 To use J-Link debugger with PSoC™ MCUs

1. Make sure you have J-Link software version 6.62 or newer.
2. Select the **Debug** tab in the Options for Target dialog, select J-LINK / J-TRACE Cortex as debug adapter, and click "Settings":

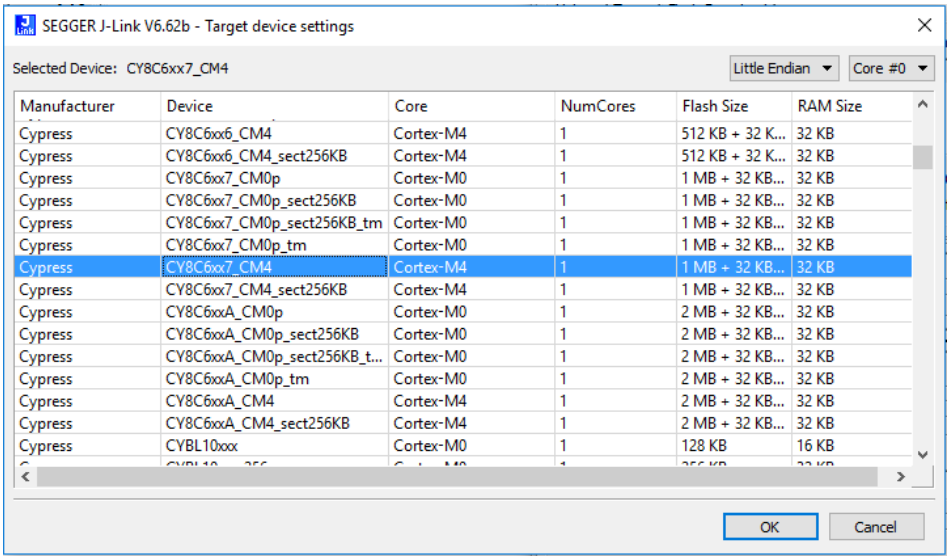


3. If you see the following message, click **OK** in the Device selection message box:



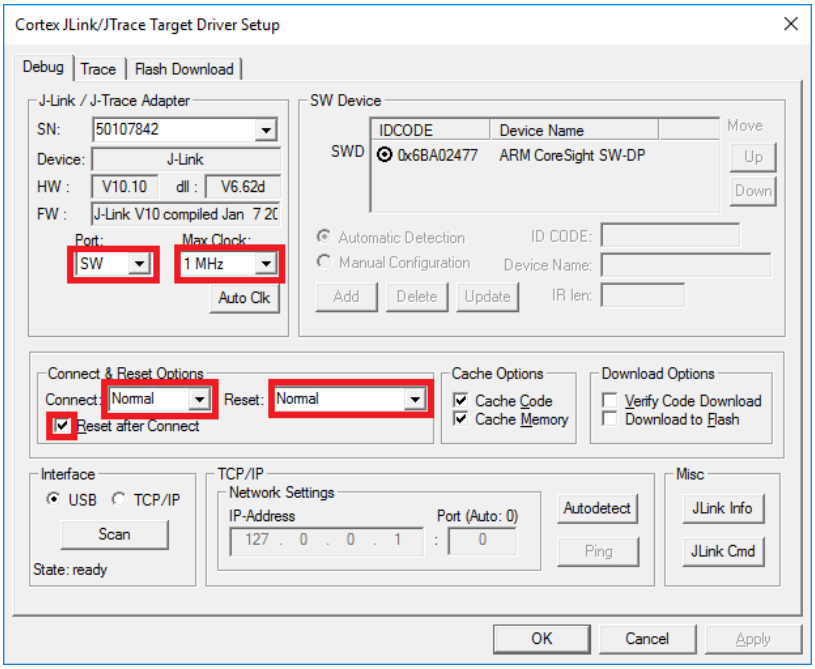
Programming/Debugging

4. Select the appropriate target in Wizard:

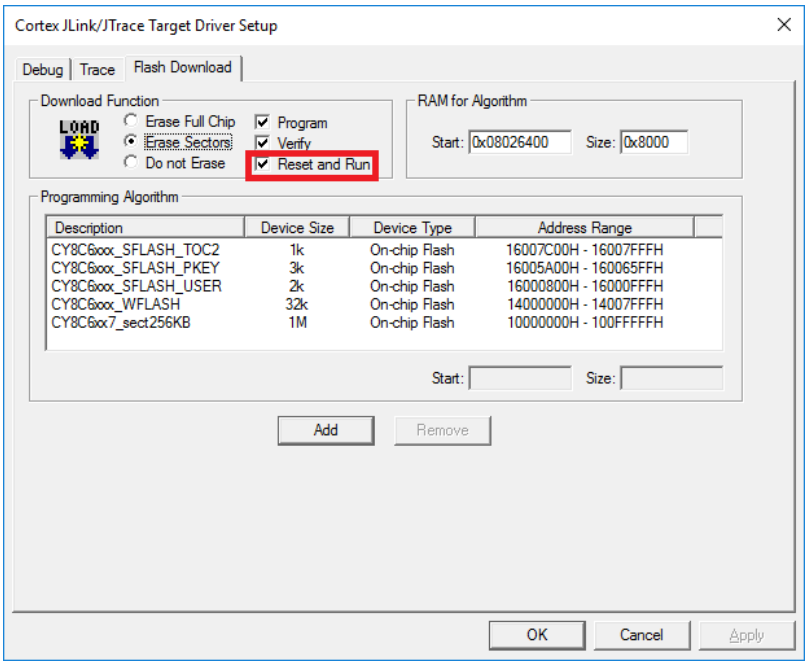


5. Go to **Debug** tab in **Target Driver Setup** dialog and select:

   - set Port to "SW"
   - set Max Clock to "1 MHz"
   - set Connect to "Normal"
   - set Reset to "Normal"
   - enable Reset after Connect option



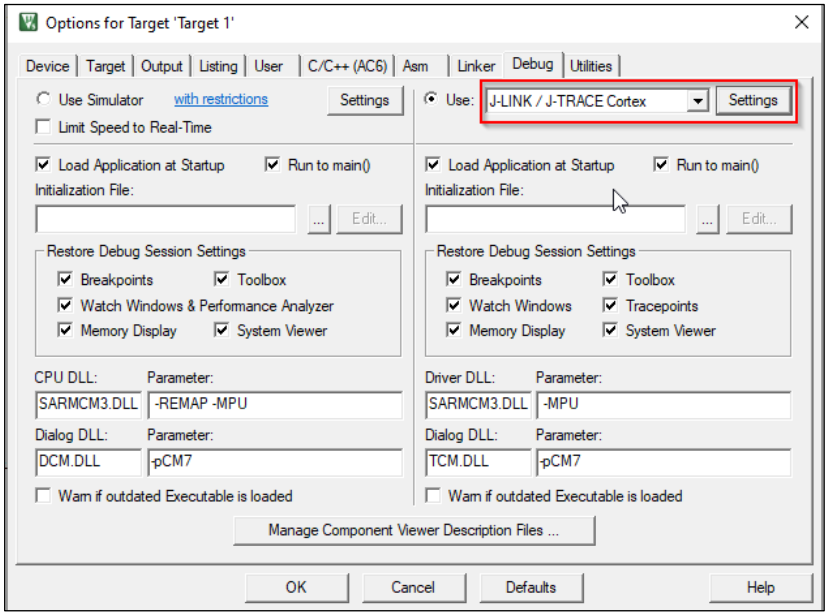
6. Select the **Flash Download** tab in **Target Driver Setup** dialog and select "Reset and Run" option after download if needed:



4.3 To use J-Link debugger with XMC7000 devices

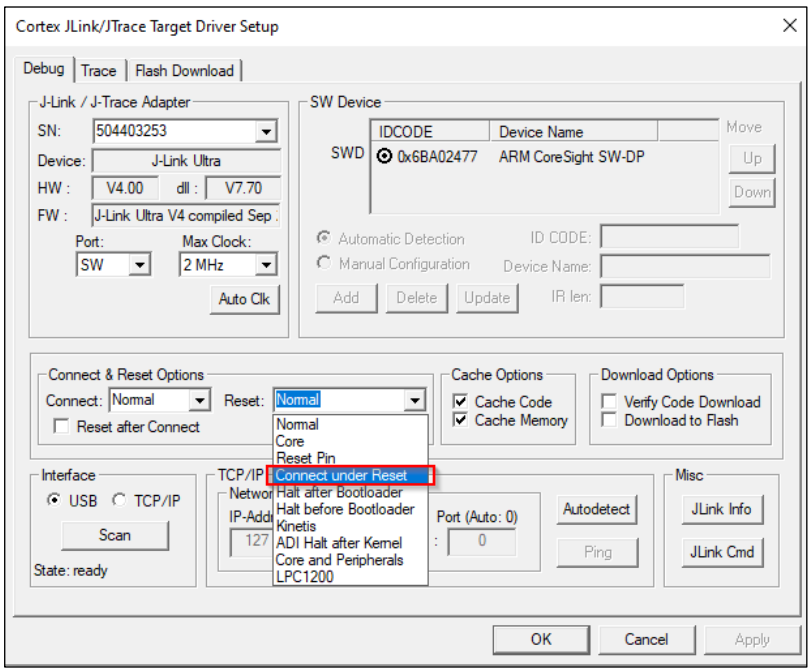
4.3.1 Program set-up instructions

1. Select the **Debug** tab in the Options for Target dialog, select "J-LINK / J-TRACE Cortex" as the debug adapter, and click **Settings**.

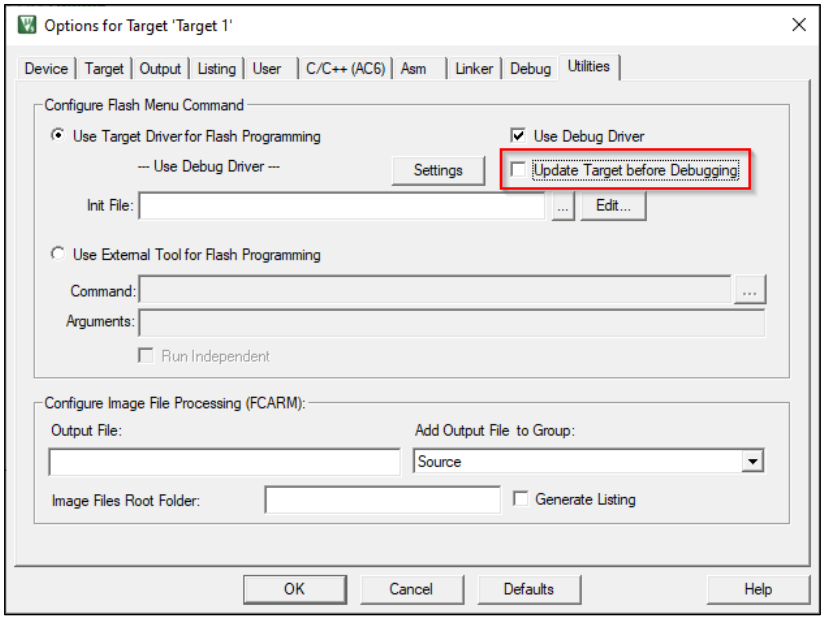


Programming/Debugging

2. On the **Debug** tab in the Cortex J-Link/JTrace Target Driver Setup dialog, select "Connect under Reset" on the **Reset** pull-down menu, and then click **OK**.

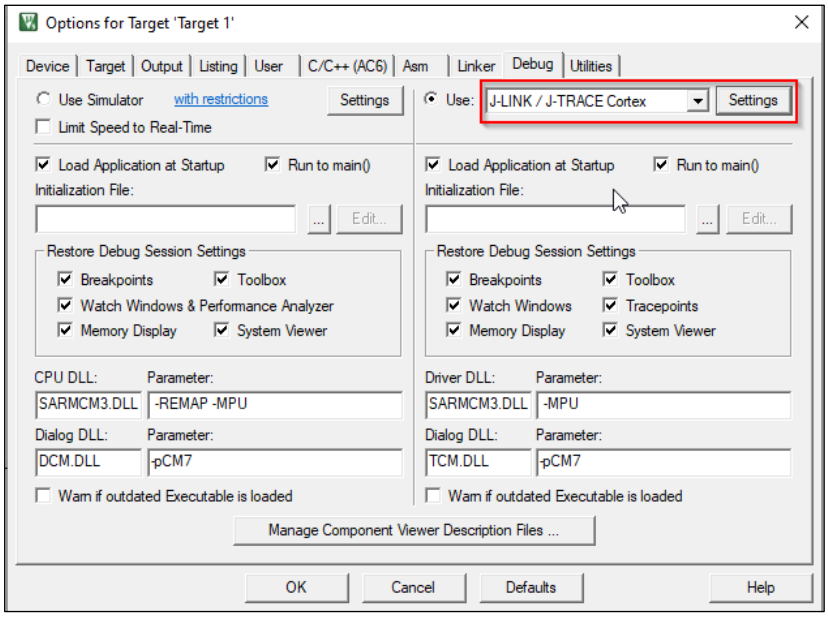


3. Select the **Utilities** tab in the Options for Target dialog and deselect the **Update target before Debugging** check box.



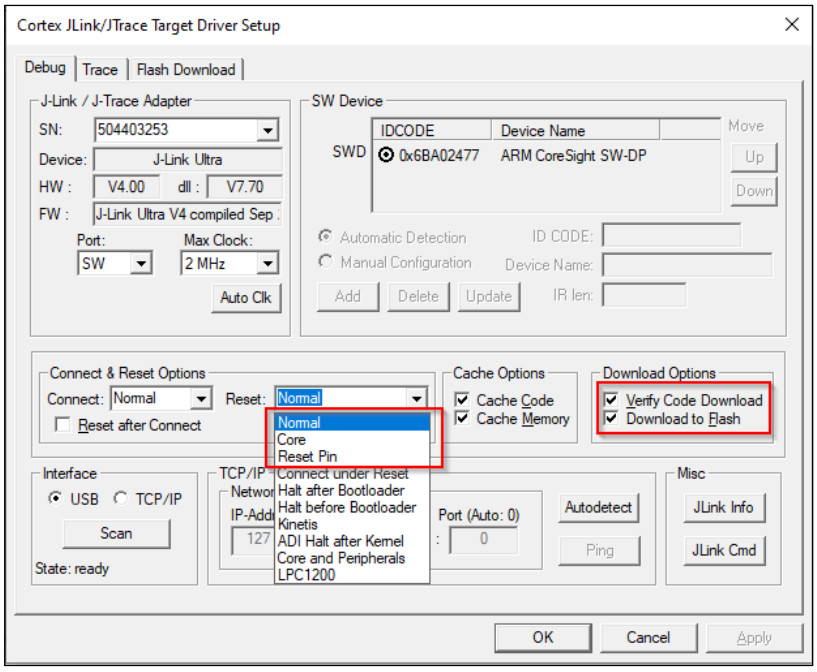
4.3.2 Debug set-up instructions

1. Select the **Debug** tab in the Options for Target dialog, select "J-LINK / J-TRACE Cortex" as the debug adapter, and click **Settings**.



2. On the **Debug** tab in the **Target Driver Setup** dialog:

   - **Reset** pull-down: select "Normal," "Core" or "Reset Pin"
   - **Download Options**: enable "Verify Code Download" and "Download to Flash"

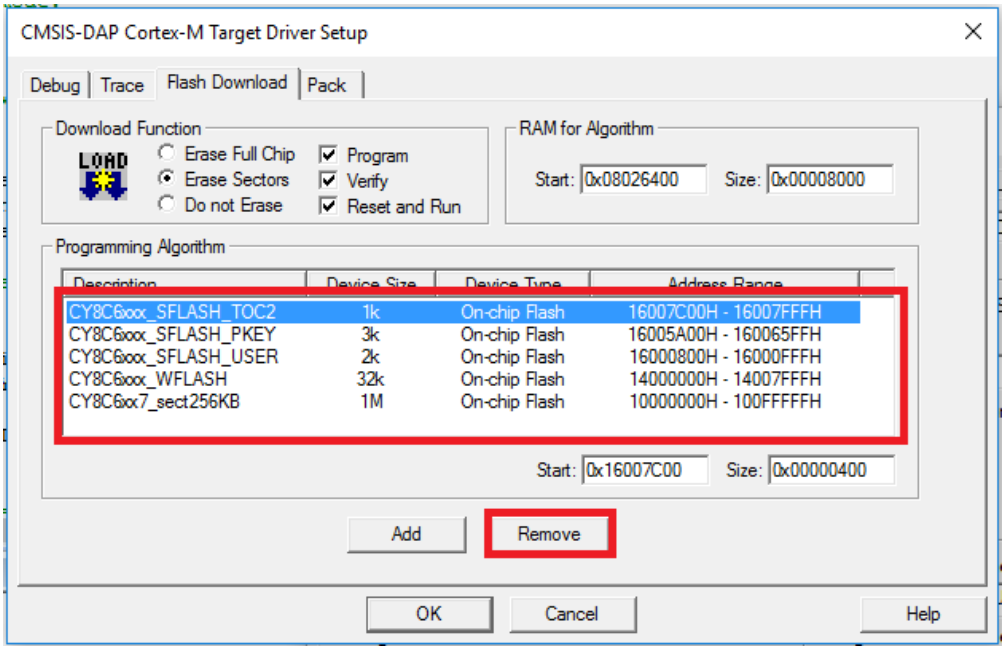


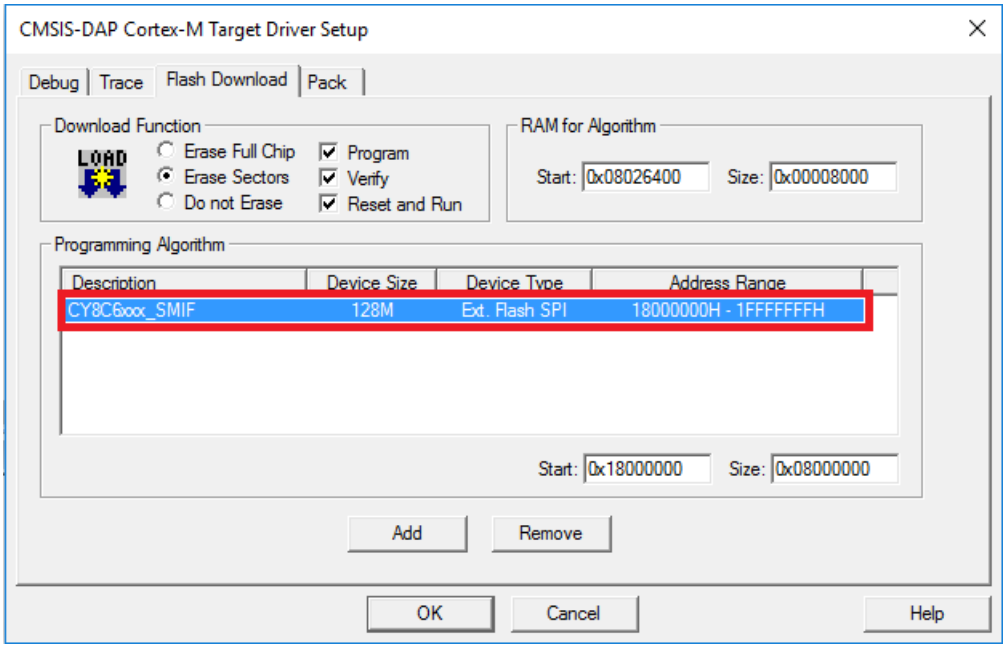
4.4 Program external memory

1. Download internal flash as described above.

   Notice "No Algorithm found for: 18000000H - 1800FFFFH" warning.

2. Select the **Flash Download** tab in **Target Driver Setup** dialog and remove all programming algorithms for On-chip Flash and add programming algorithm for External Flash SPI:





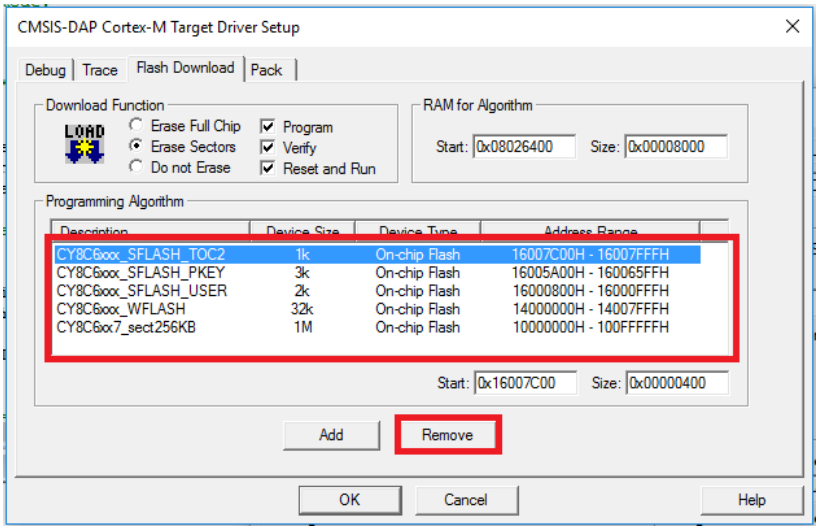
3. Download flash.

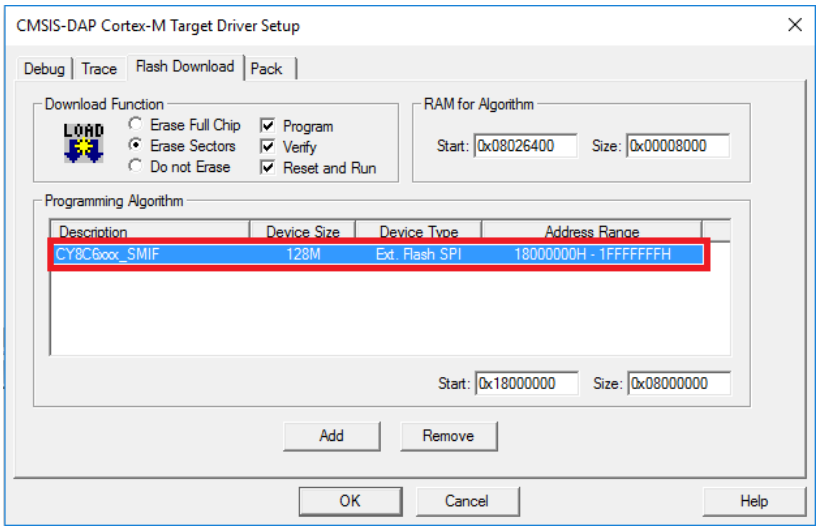
   Notice warnings:

   - No Algorithm found for: 10000000H - 1000182FH
   - No Algorithm found for: 10002000H - 10007E5BH
   - No Algorithm found for: 16007C00H - 16007DFFH

4.5 Erase external memory

1. Select the **Flash Download** tab in **Target Driver Setup** dialog and remove all programming algorithms for On-chip Flash and add programming algorithm for External Flash SPI:





2. Click **Flash > Erase** in menu bar.

5 Multi-core debugging

This section explains how to set up multi-core debugging in the µVision IDE.

5.1 Supported debugger probes

- KitProg3 onboard programmer
- MiniProg4
- ULINK2
- J-Link

5.2 Opening µVision multi-core projects

After you create a ModusToolbox™ multi-core application for use with µVision, do the following:

1. Navigate to ModusToolbox™ project directory for one of the cores and double-click project description file (either *.cprj or *.cpdsc depending on µVision version).

   The first time you do this, a dialog may pop-up offering to install missing CMSIS-Pack. Click **Yes** and the pack will be installed.

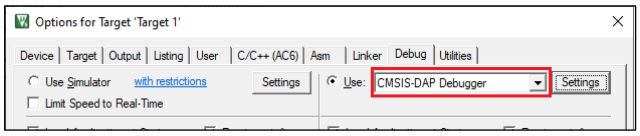
2. Repeat the same process for the CM4/CM7 project(s). This will create and open µVision projects.

5.3 Debugger configuration

Next, configure projects to launch multi-core debugging.

5.3.1 Configure CM0+ project

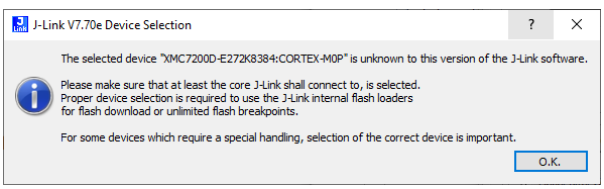
1. Go to **Project > Options for Target <target_name>**, switch to the **Debug** tab, select the applicable debug probe (CMSIS-DAP or J-Link) as shown:



   - If using ULINK2, select the **CMSIS-DAP Debugger** option as the debug probe, because the ULINK2 driver does not support multi-core debugging.

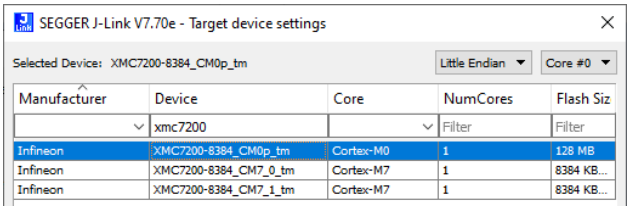
2. Click the **Settings** button to configure the target driver.

   - If you select the J-Link probe, a pop-up window might display reporting that the device is unknown to J-Link software.

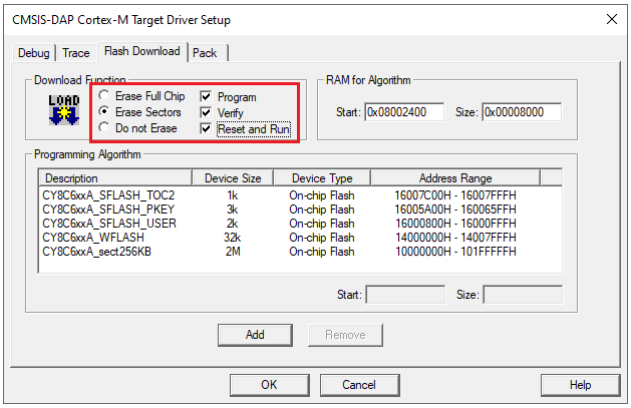


   - If so, click ***O.K.*** and select the device manually in the opened Target device settings dialog. For XMC7200 devices, there will be three aliases, each dedicated to a separate core.

Multi-core debugging



3. Switch to the **Flash Download** tab, select the **Erase Sectors** radio button, and select the **Program**, **Verify**, and **Reset and Run** check boxes.



4. Click **OK** to close the Target Driver Setup dialog.

5. Next, configure the project so that it also programs other image(s) from the CM4/CM7 project(s). Do this using the *.ini file.

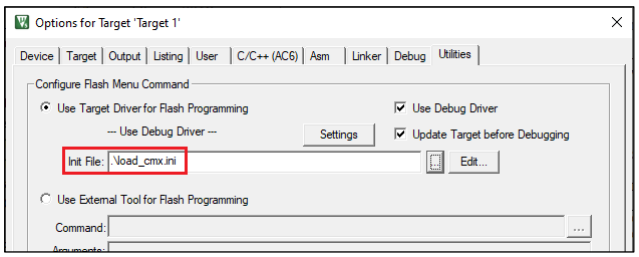
   a. Create a new empty file named *load_cmx.ini* and save it inside the CM0+ project directory.

   b. Add a `LOAD` command with a path to the CM4/CM7 images. For example:

   ```
   LOAD "..\\proj_cm4\\proj_cm4_Objects\\proj_cm4.axf"
   ```

   c. Add as many `LOAD` commands for all the CM4/CM7 projects as you have.

6. Go to **Project > Options for Target <target_name>**, select the **Utilities** tab, and specify the created *load_cmx.ini* file in the **InitFile** edit field.

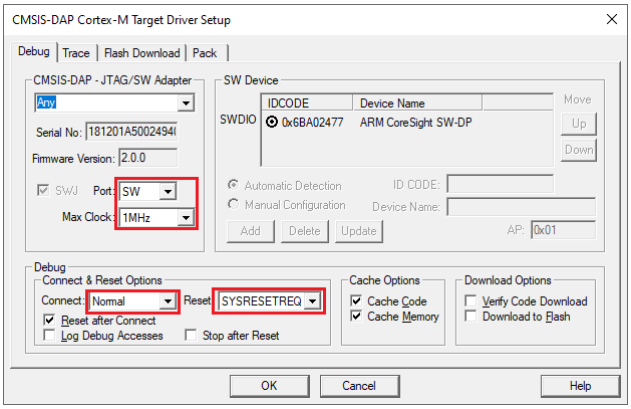


7. Switch to the **Debug** tab, and click the **Settings** button.

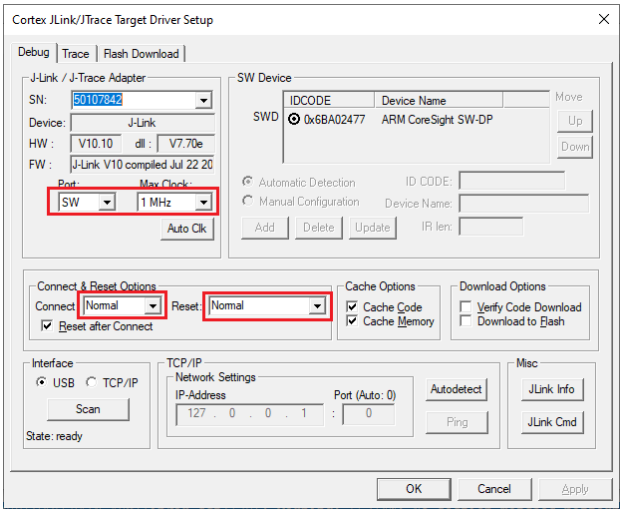
   The configuration settings are different for CMSIS-DAP/ULINK2 and J-Link. Refer to the following sections for the applicable options:

   - **CMSIS-DAP/ULINK2 Target Driver Setup** – Use the following options:
     - **Port**: SW
     - **Max Clock**: 1 MHz
     - **Connect**: Normal
     - **Reset**: SYSRESETREQ

Multi-core debugging



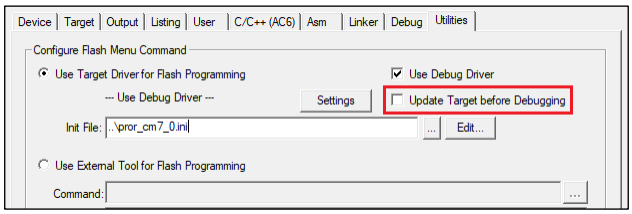
- **J-Link Target Driver Setup** – Use the following options:
  - **Port**: SW
  - **Max clock**: 1 MHz
  - **Connect**: Normal
  - **Reset**: Normal



That completes configuring the CM0+ project. The next step is to configure CM4/CM7 project(s).

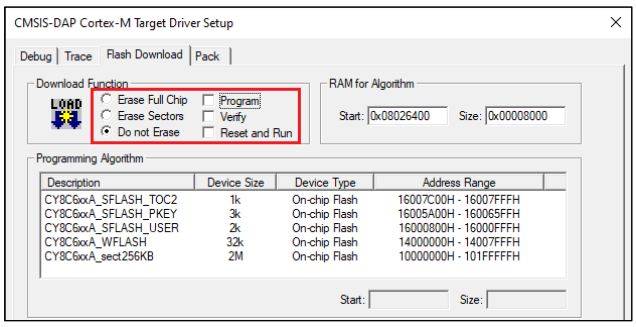
5.3.2 Configure CM4/CM7 project

1. Go to **Project > Options for Target <target_name>**, switch to the **Utilities** tab and deslect the Update Target before Debugging check box.



2. Switch to the **Debug** tab, select the applicable debug probe (CMSIS-DAP or J-Link).
   - If using ULINK2, select the **CMSIS-DAP Debugger** option as the debug probe, because the ULINK2 driver does not support multi-core debugging.
3. Click the **Settings** button to configure the target driver.

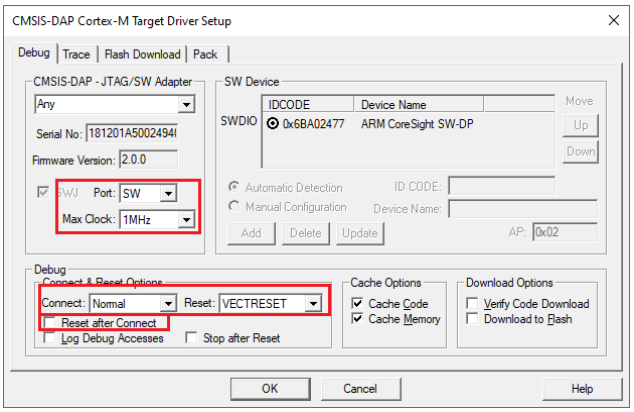
4. On the Target Driver Setup dialog, switch to the **Flash Download** tab, select the **Do not Erase** radio button, and deselect the **Program**, **Verify**, and **Reset and Run** check boxes.



5. Switch to the **Debug** tab.

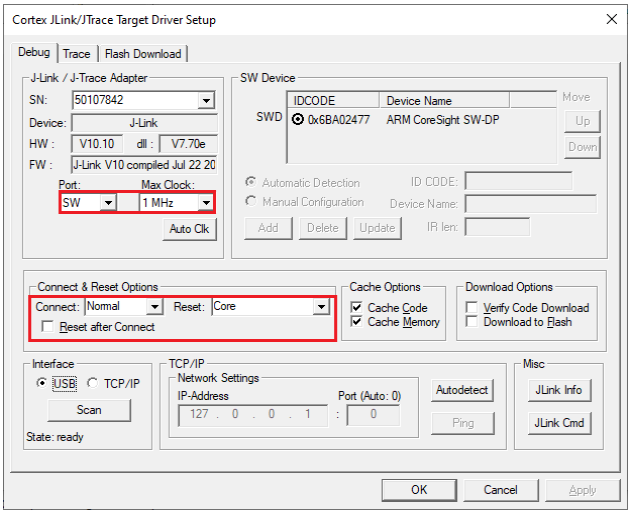
   The configuration settings are different for CMSIS-DAP/ULINK2 and J-Link. Refer to the following for the appropriate options:

   - **CMSIS-DAP/ULINK2 Target Driver Setup** – Use the following options:
     - **Port**: SW
     - **Max Clock**: 1 MHz
     - **Connect**: Normal
     - **Reset**: VECTRESET
     - **Reset after Connect** check box: deselected



   - **J-Link Target Driver Setup** – Use the following options:
     - **Port**: SW
     - **Max Clock**: 1 MHz
     - **Connect**: Normal
     - **Reset**: Core
     - **Reset after Connect** check box: deselected

Multi-core debugging



6. Click **OK** to close the Target Driver Setup dialog.
7. Save the project(s).

5.4 Launching multi-core debug session

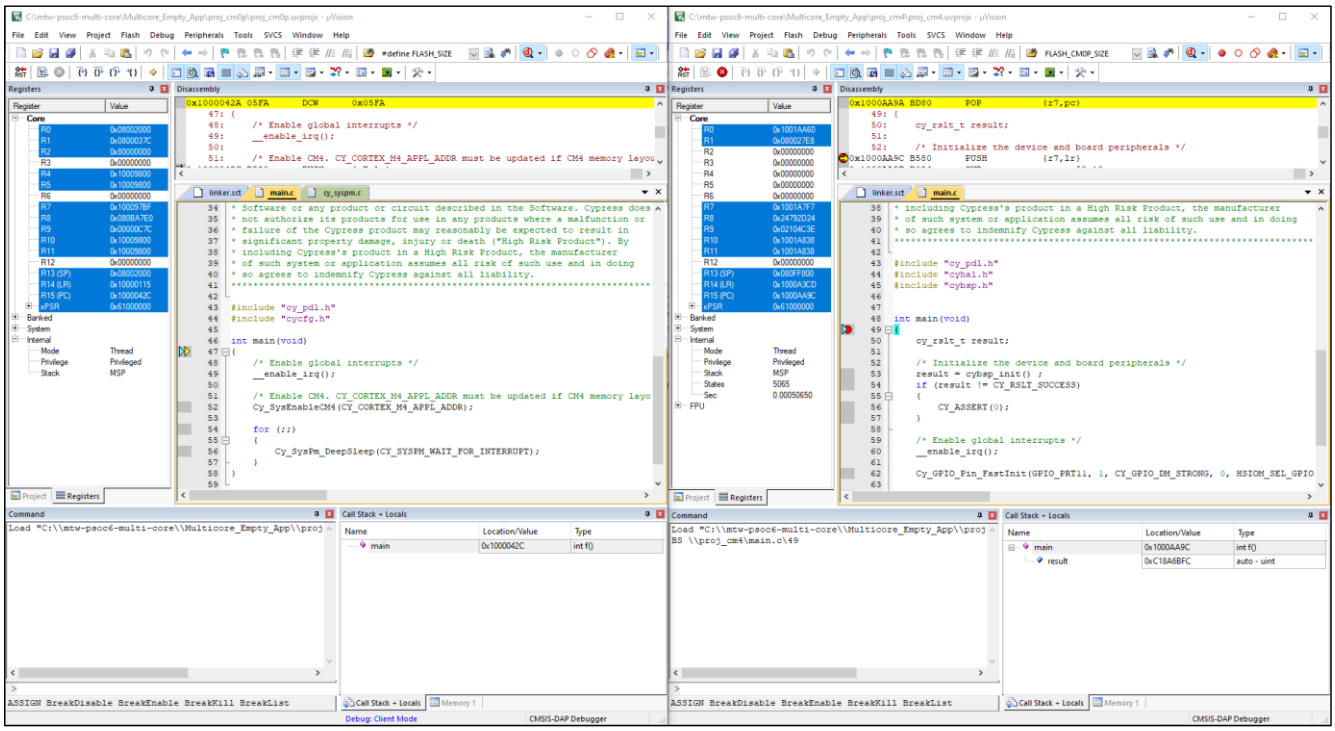
To launch a multi-core debug session, all your µVision projects must be opened in separate IDE instances.

1. Open a µVision IDE session with the project for the CM0+ core and start debugging by pressing **Debug > Start/Stop Debug Session**. This will program all images, reset the target, and halt at the beginning of the CM0+ project `main()`.
2. Repeat the same process for the CM4/CM7 core(s). This will attach the running CM4/CM7 core that will be spinning in the boot code until the CM0+ project starts it.

Note: Ensure both projects are built before launching a debug session.

Multi-core debugging

For dual-core MCUs, the projects will appear similar to these images:



The left side of the screen shows a µVision IDE instance attached to the CM0+ core. The right side shows the CM4 core has not started yet. Once the `Cy_SysEnableCM4()` function on the CM0+ core has been executed, the CM4 will start executing its application. You can step through the code by switching back and forth between the two µVision IDE instances.

6 Patched flashloaders for AIROC™ CYW208xx devices

To enable support for different QSPI settings, the ModusToolbox™ QSPI Configurator patches flashloaders and stores files for them in the application directory. When exporting such applications to Keil µVision, these patched flashloader files must be copied into the appropriate directory.

1. Copy the *CYW208xx_SMIF.FLM* file.located in the *<app-dir>\libs\<Kit-Name>\COMPONENT_BSP_DESIGN_MODUS\GeneratedSource* directory.
2. Paste the flashloader file to the *C:\Program Files\IAR Systems\Embedded Workbench 9.0\arm\config\flashloader\Infineon\CYW208XX* directory.
3. Also, to use the SEGGER J-Link debugger, paste the *CYW208xx_SMIF.FLM* file to the *C:\Program Files\SEGGER\J-Link\Devices\Cypress\cat1b* directory.

Revision history

Revision	Date	Description
**	2023-05-15	New document.
*A	2023-06-02	Removed obsolete instructions for customizing linker scripts.

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2023-06-02

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-37570 Rev. *A

Important notice

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie")

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.