Keil µVision for ModusToolbox™ user guide

ModusToolbox™ tools package version 3.8.0

About this document

A newer version of this document may be available on the web here.

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.

The general flow for working with an Infineon device in Keil µVision includes:

- Download/install software
- Create ModusToolbox™ application using Project Creator
- Create project(s) in Keil µVision
- Configure and Build projects
- Program the device
- Debug the application

Document conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, menus and sub-menus.
Italics	Denotes file names and paths.
`Monospace`	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™ software 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

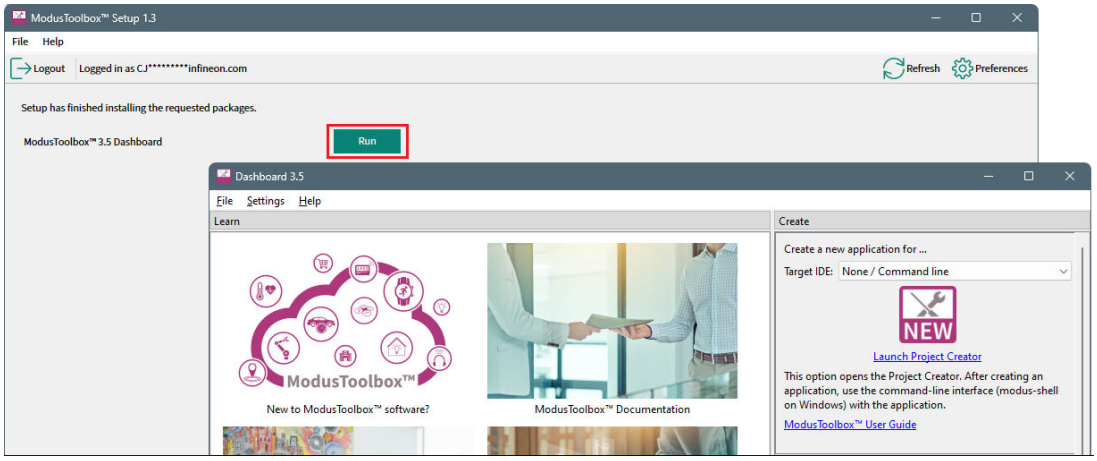
Table of contents

1 Download/install software

1.1 ModusToolbox™ software

Download the ModusToolbox™ Setup program from https://softwaretools.infineon.com/tools/com.ifx.tb.tool.modustoolboxsetup. Refer to the instructions in the ModusToolbox™ software installation guide for how to install the necessary ModusToolbox™ tools and packages.

Once installation of all the tools is complete, click **Run** to launch the Dashboard.



1.2 Keil µVision (Windows only)

We recommend and have tested Keil µVision (MDK-ARM) version 5.43 for all ModusToolbox™ supported devices.

***Note**: Do not use version 5.42a. There is a known defect with cprj files in that version. Also, do not use version 5.43a due to an issue with FPU settings.*

The default installation location for the ARM compiler is *C:\Keil_v5\ARM*. Set the `CY_COMPILER_ARM_DIR` environment variable to the correct installation path using forward slashes. Alternatively, you can set this variable in the *Makefile* for each application. Refer to the ModusToolbox™ tools package user guide for more details about build system variables.

1.3 J-Link

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

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

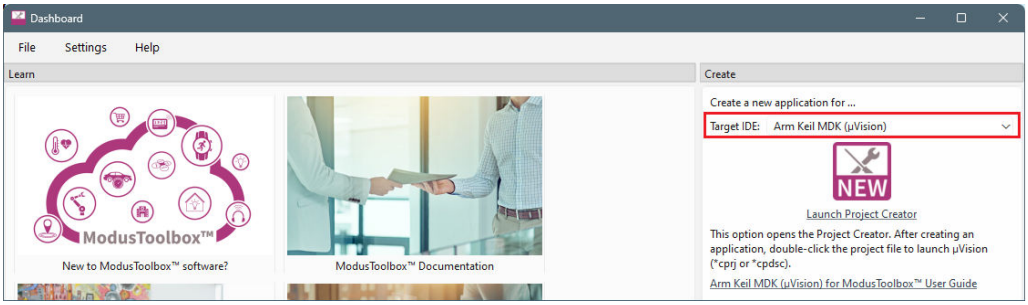
2 Create/export application for Keil µVision

This section covers the ways to get started using Keil µVision with ModusToolbox™ software.

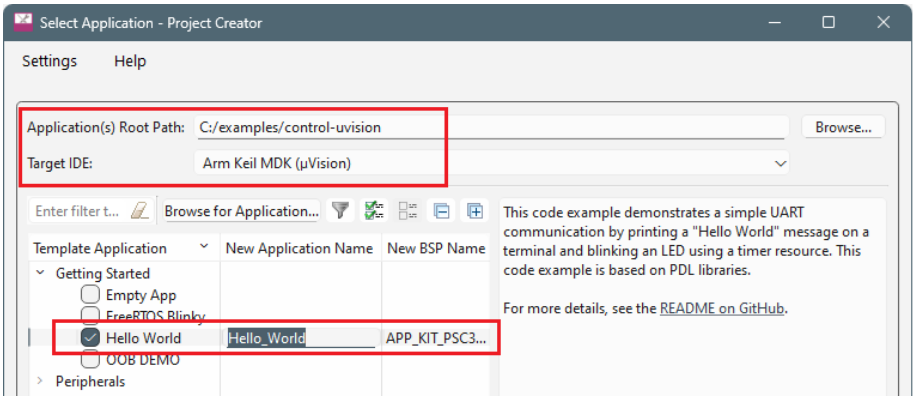
2.1 Create/export ModusToolbox™ application

Create new application

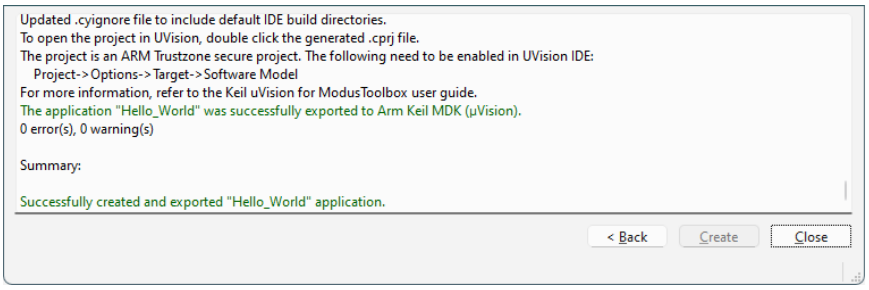
1. Use the Dashboard to open the Project Creator tool and create a ModusToolbox™ application for Keil µVision. Refer to the Dashboard user guide for more details for this tool.



2. In the Project Creator tool, make sure to select the **Application(s) Root Path** and that the **Target IDE** is set to Arm Keil MDK (µVision), in addition to choosing the **Template Application**. Refer to the Project Creator user guide for more details.



3. After the project creation process completes, check the messages in the console.



4. Close the Project Creator and the Dashboard. Proceed to the section in this document for your device family.

Export existing application

Instead of creating a new 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 Keil µVision. Open a terminal window (modus-shell in Windows) and type the following:

```
make uvision CY_IDE_PRJNAME=[project-name] TOOLCHAIN=ARM
```

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

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 in 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.

Check the output log for instructions.

Postbuild script

The process to create or export an application for use in Keil µVision creates a script file named *ide_postbuild.bat*. This script updates the hex file as needed for security and multi-core support. It requires you to set the `CY_COMPILER_ARM_DIR` environment variable. For example:

```
CY_COMPILER_ARM_DIR=C:/Keil_v5/ARM/ARMCLANG
```

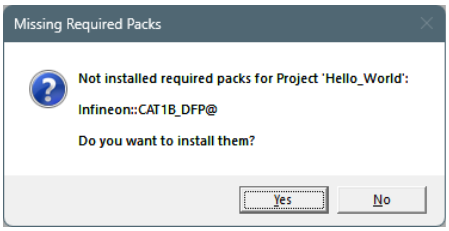
Use forward slashes (even on Windows) because of UNIX/Linux path rules.

2.2 Create Keil µVision project(s)

Creating or exporting the application generates a **.cprj* file in the application/project directory. The *cprj* file extension should have the association enabled to open it in Keil µVision.

1. Double-click the **.cprj* file. This launches the Keil µVision IDE.

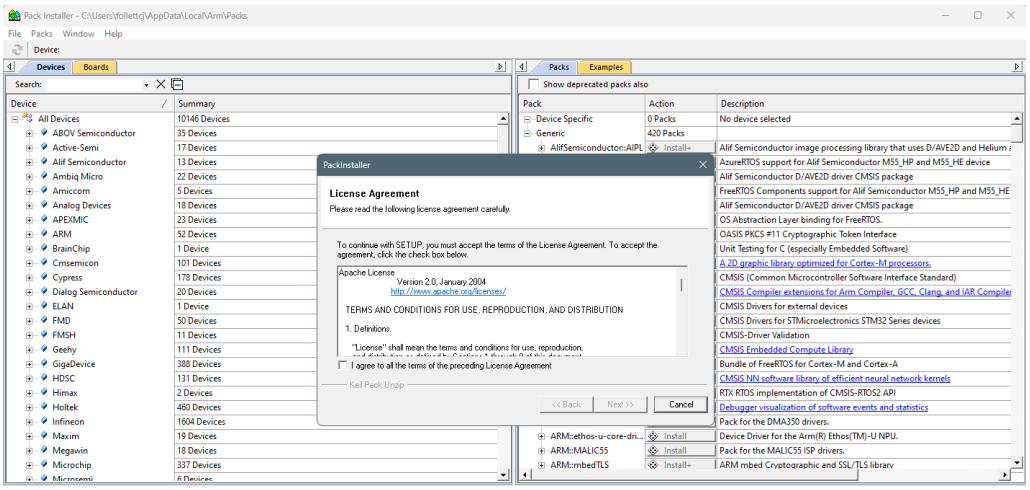
The first time you do this, a dialog similar to the following displays:



2. Click **Yes** to install the device pack. You only need to do this once per device type.

3. Follow the steps in the Pack Installer to properly install the device pack.

2 Create/export application for Keil µVision



***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 the application will be created for you in the IDE.

5. If you're working with a multi-core or multi-project application, a *cprj* file is created in each sub-project directory. Double-click the **.cprj* file in each folder (for example, PSOC™ Edge: cm33s, cm33ns, and cm55, XMC7xxx: cm0p, cm7_0, and cm7_1).

3 Miscellaneous notes

This chapter contains miscellaneous notes that may be applicable to any device:

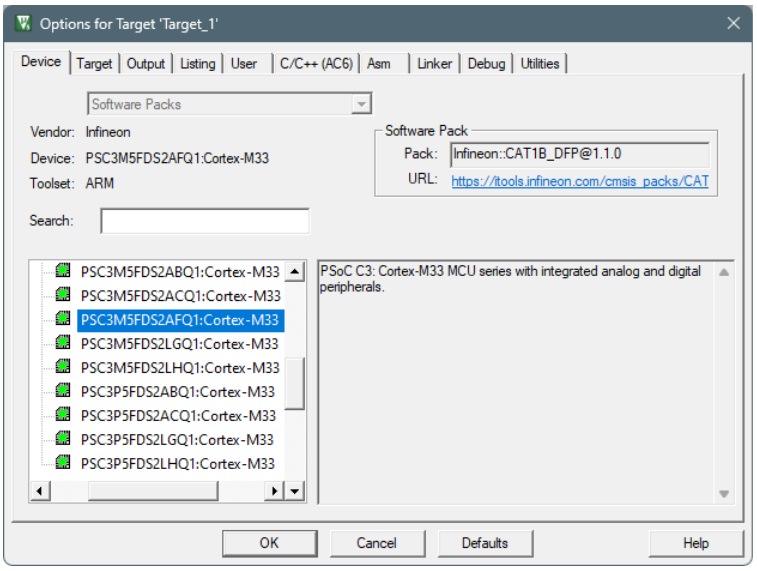
- Supported debugger probes
- To use KitProg3/MiniProg4, CMSIS-DAP, and ULINK2 debuggers
- To use J-Link debugger
- Suppress linker build warnings
- Empty defines
- Patched flashloaders
- Perform ETM/ITM trace

3.1 Supported debugger probes

- KitProg3 on-board programmer
- MiniProg4
- ULINK2 in CMSIS-DAP mode
- J-Link

3.2 To use KitProg3/MiniProg4, CMSIS-DAP, and ULINK2 debuggers

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

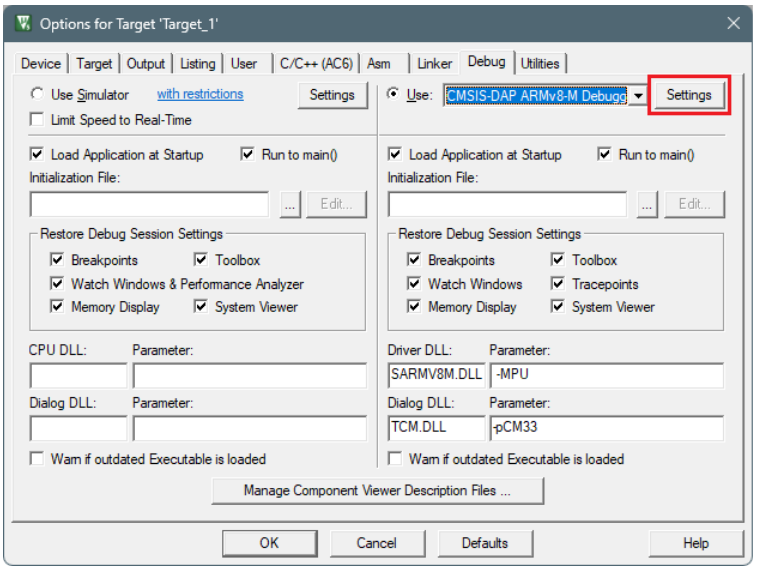


2. Select the **Debug** tab.

   ***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).*

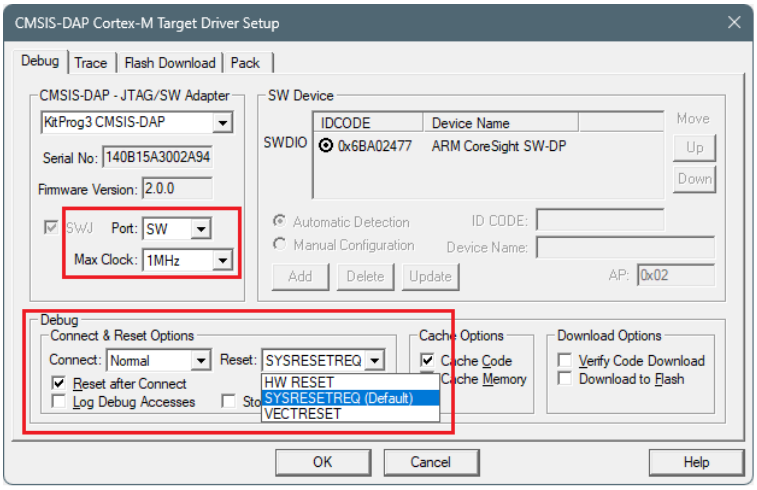
3. Click the **Settings** button.

3 Miscellaneous notes



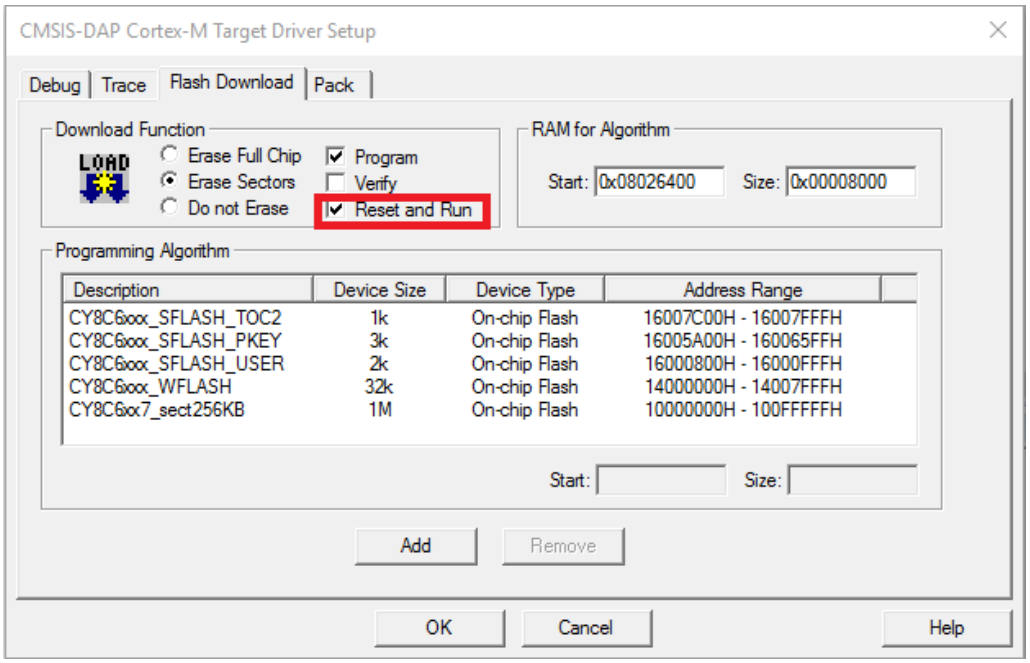
4. On the Target Driver Setup dialog on the **Debug** tab, select the following:

- 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"
  - For other devices, leave as is.
- enable **Reset after Connect** option

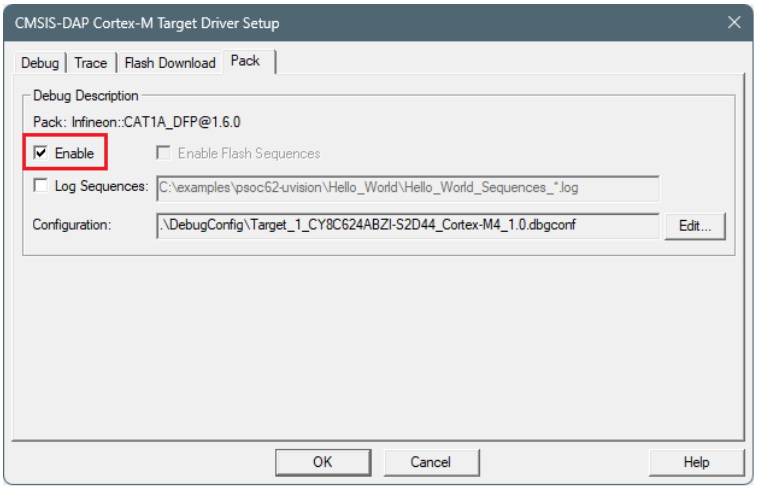


5. Select the **Flash Download** tab and select "Reset and Run" option after download, if needed:

3 Miscellaneous notes



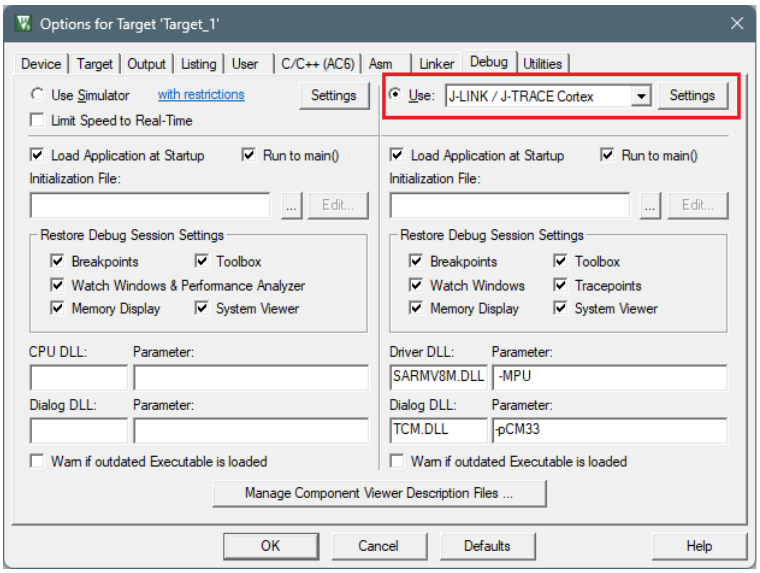
6. Select the **Pack** tab and check if the appropriate "DFP" is enabled:



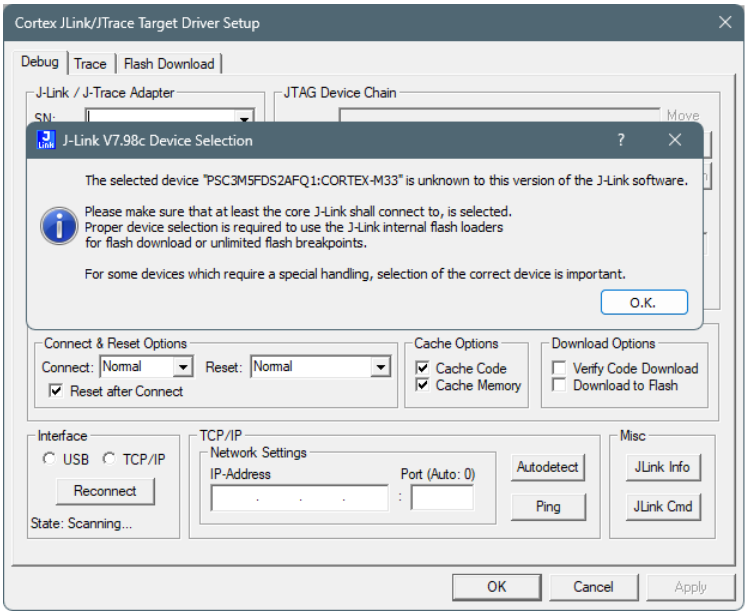
3.3 To use J-Link debugger

1. Make sure you have J-Link software version 8.12 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**:

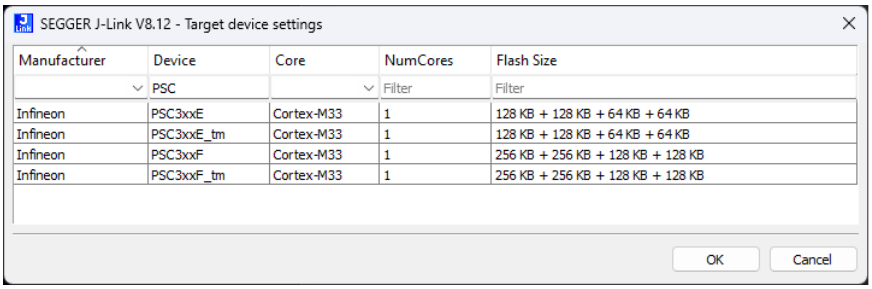


3. If you see the following message, click **OK** in the Device selection message box:



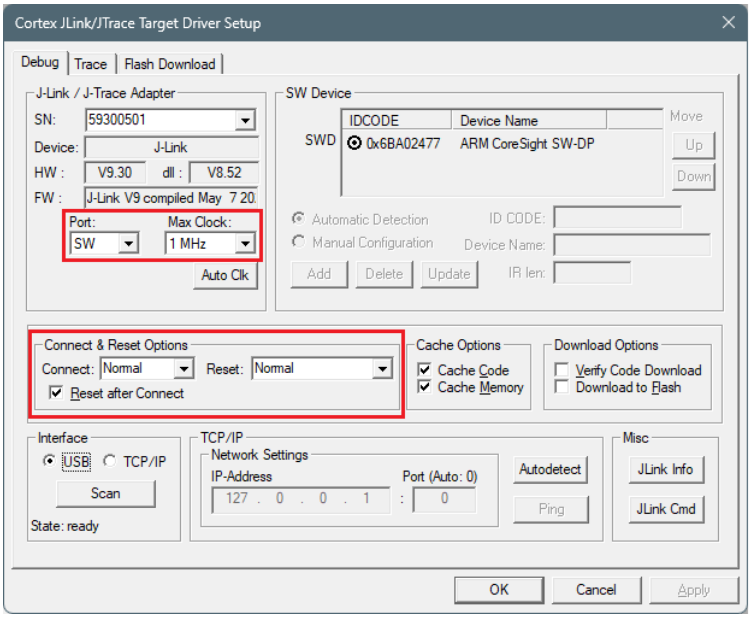
4. Select the appropriate device in the Target device settings and click **OK**.

3 Miscellaneous notes

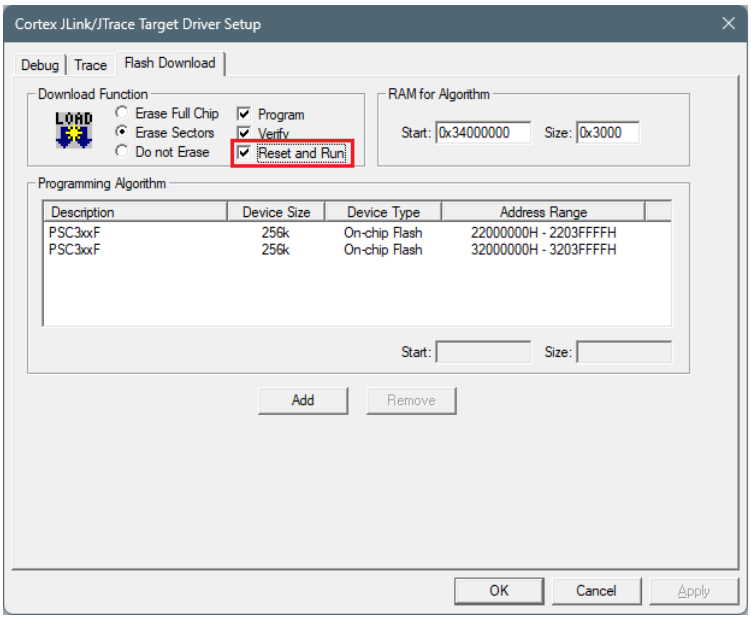


5. Go to the **Debug** tab in Target Driver Setup dialog and select:

- 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:

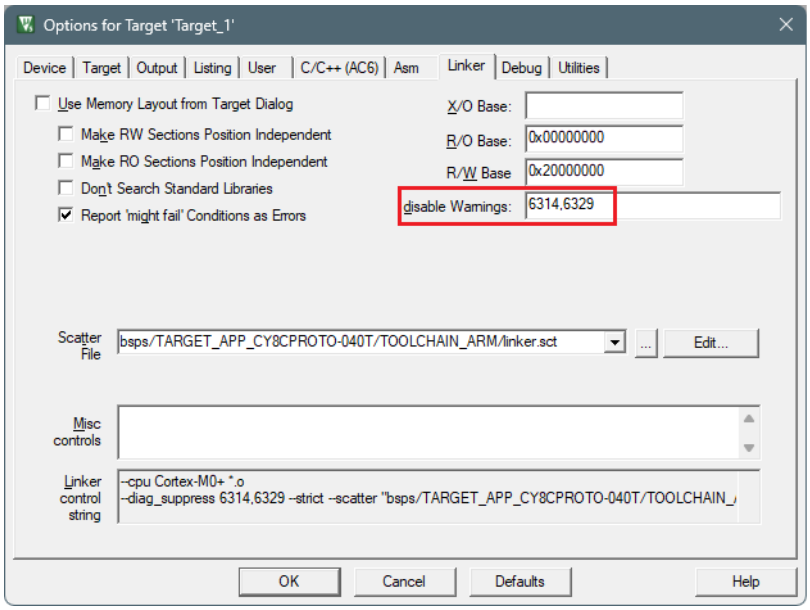


3.4 Suppress linker build warnings

Often when building an application in uVision, you will see various warnings about unused sections defined in the linker scripts, such as:

```
warning: L6314W: No section matches pattern EXCLUDE(*cy_smif.*).
```

To suppress these warnings, add "6314,6329" to the **disable Warnings** setting in the Options **Linker** tab.



3.5 Empty defines

There is a known issue in Keil µVision with empty defines. When you try to build an application, you may see an error similar to this:

```
error: You now need to define either
FIXED_POINT or FLOATING_POINT
```

To fix this, make sure empty defines are quoted. For example:

```
DEFINES+=FLOATING_POINT EXPORT=" " USE_CMSIS_DSP \
```

3.6 Patched flashloaders

Step 1: Identify the CMSIS Pack and FLM file

- Determine the name and version of the CMSIS Pack containing the FLM file you want to overwrite (for example, *AIROC_DFP.1.2.0*).
- Identify the name of the patched flashloader (the FLM file) you want to overwrite (for example, *CYW208xx_SMIF.FLM*). This file is located in the *<app-dir>\bsps\<Kit-Name>\config\GeneratedSource* directory.

Step 2: Locate the destination FLM file

- Open File Explorer and navigate to the directory: *%LocalAppData%\Arm\Packs\<Pack_Name>\<Version>\Flash*
- Replace *<Pack_Name>* with the name of the CMSIS Pack (for example, *Infineon\AIROC_DFP*) and *<Version>* with the version number of the CMSIS Pack (for example, *1.2.0*).
- In this directory, find the file named *<FLM_File_Name>.FLM* (for example, *CYW208xx_SMIF.FLM*).

Step 3: Overwrite the FLM file

- Ensure that Keil µVision and any other applications using the FLM file are closed.
- Replace the destination *<FLM_File_Name>.FLM* file with the patched one you want to use.
- Make sure to keep the same file name and extension (.FLM) to avoid any issues.

***Note**: When using J-Link with µVision, the flashloaders will be taken from the CMSIS DFP. So, there is no need to replace flash loaders in SEGGER J-Link software.*

3.7 Perform ETM/ITM trace

For PSOC™ Control and PSOC™ Edge devices, refer to application note AN242338 - Performing ETM and ITM trace on PSOC™ Control C3 and PSOC™ Edge E8 MCUs.

For PSOC™ 6 devices, refer to application note AN235279 - Performing ETM and ITM trace on PSOC™ 6 MCU.

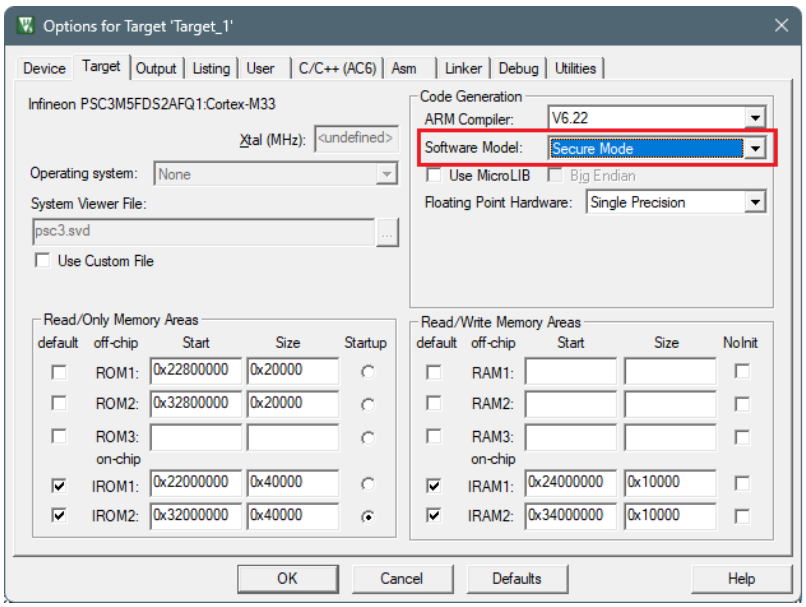
4 PSOC™ Control C3 application

Follow instructions in Create/export application for Keil µVision. At the end of the process, you should see messages in the console. For a PSOC™ Control C3 device, one of the messages should read as follows:

```
The project is an ARM Trustzone secure project. Check the uVision IDE Project->Options->Target-
>Software Model to ensure "Secure" is enabled.
or
The project is an ARM Trustzone non-secure project. Check the uVision IDE Project->Options-
>Target->Software Model to ensure "Non-Secure" is enabled.
```

In most cases, PSOC™ Control C3 applications are set to Trust Zone Secure by default. For more details about TrustZone technology, refer to the Arm® website: https://www.arm.com/technologies/trustzone-for-cortex-m.

When you open the application in Keil µVision, you need to check the TrustZone setting. Open the Options for Target dialog, select the **Target** tab, and ensure the **Software Model** option is set to "Secure Mode".



Save the application and select **Project > Build Target** to build it. The Output should display the progress, ending with text similar to this:

```
linking...
bsps/TARGET_APP_KIT_PSC3M5_EVK/TOOLCHAIN_ARM/linker_s_flash.sct(107): warning: L6329W: Pattern
*(.cy_sharedmem) only matches removed unused sections.
bsps/TARGET_APP_KIT_PSC3M5_EVK/TOOLCHAIN_ARM/linker_s_flash.sct(113): warning: L6314W: No
section matches pattern *(Veneer$$CMSE).
Program Size: Code=16902 RO-data=1438 RW-data=312 ZI-data=64844
Finished: 0 information, 2 warning and 0 error messages.
".\Hello_World_Objects\Hello_World.axf" - 0 Error(s), 2 Warning(s).
Build Time Elapsed:  00:00:22
```

In addition to the TrustZone technology from Arm, PSOC™ Control C3 devices have various security life cycle stages (LCS). For more details about security, refer to Application Note AN240106 - Getting started with PSOC™ Control C3 security.

The following sections provide details about working with a device with the default out of the box policy versus a device that has been provisioned.

4.1 Device with default policy

Devices are shipped with a default policy, so you can develop and debug your application repeatedly without any knowledge about security or code signing. This is also referred to as Development LCS. In order to create a valid hex file for programming and debugging in this state, we must configure the application

1. Create a *program.ini* file in the root directory of the project with the following content:

```
LOAD $L%L 0x20000000
```

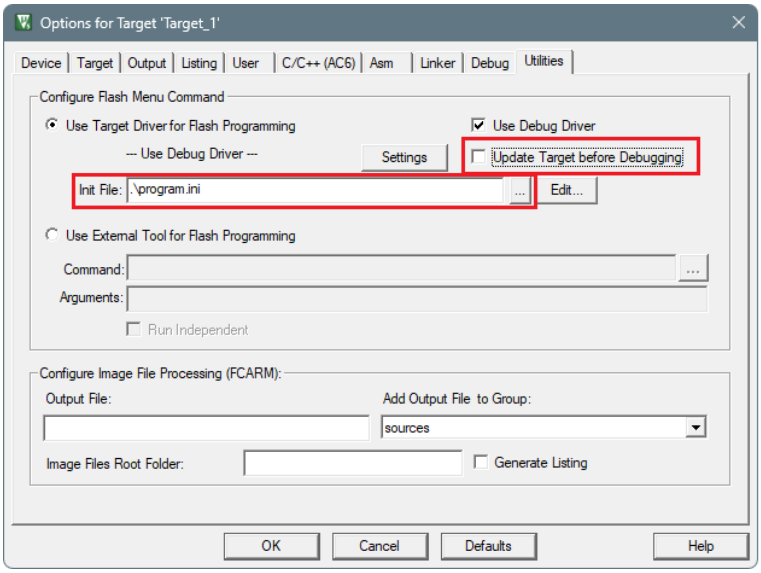
The primary purpose of this is to provide a `LOAD` command needed for programming. The `0x20000000` part shifts the application image from the C-Bus region to the S-Bus region. This is done because the ARM compiler attempts to copy the Load Region to the Execution Region, which is read-only.

2. Create a *debug.ini* file in the root directory of the project with the following content:

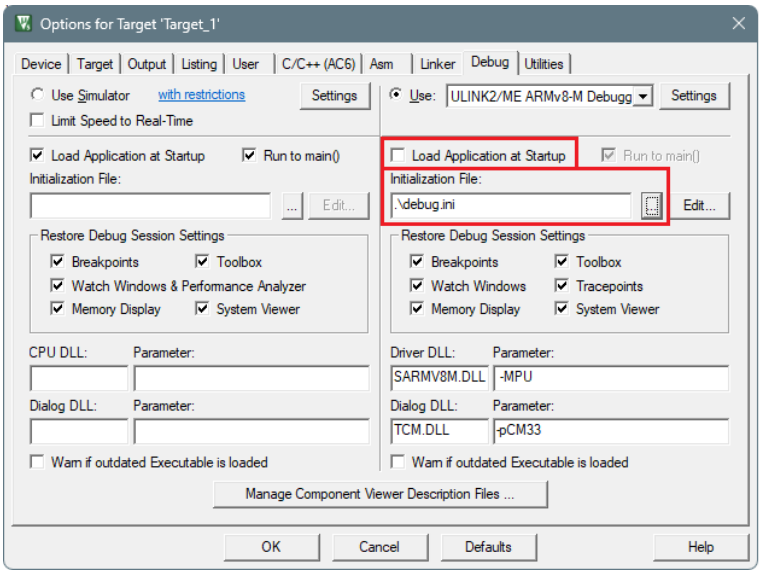
```
LOAD $L%L NOCODE CLEAR INCREMENTAL
g, main
```

This provides a `LOAD` command needed for debugging.

3. Open the Options dialog to the **Utilities** tab, deselect the **Update Target before Debugging** check box, and select the *program.ini* file in the **Init File** field:



4. Switch to the **Debug** tab. Make sure that the **Load Application at Startup** check box is not checked.
5. Select the *debug.ini* file in the **Initialization File** field:



6. Click **OK** to close the Options dialog.
7. Select **Project > Build Target** to build the application and execute post-build commands.
8. Configure the applicable debugger settings. See Miscellaneous notes.
9. To program the device, select **Flash > Download**.
10. To start the debugger, select **Debug > Start/Stop Debug Session**.

4.2 Provisioned device

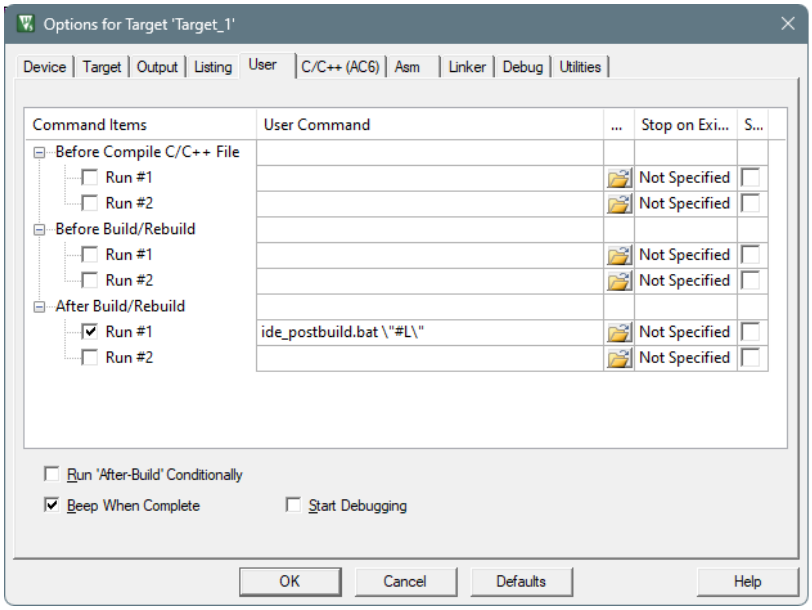
If you plan to work with a provisioned device, use the Basic Secure App. This example automates the signing steps through the *ide_postbuild.bat* script, which is generated automatically when you export the project (see Create/export ModusToolbox™ application).

Change the *program.ini* file or create a new one with the following content:

```
LOAD $L\..\build\APP_KIT_PSC3M5_EVK\Debug\mtb-example-ce240783-secureboot.hex
```

Open the Options dialog to the **User** tab and enable the **Run #1** check box under **After Build/Rebuild**. Then, enter `ide_postbuild.bat \"#L\"`.

4 PSOC™ Control C3 application



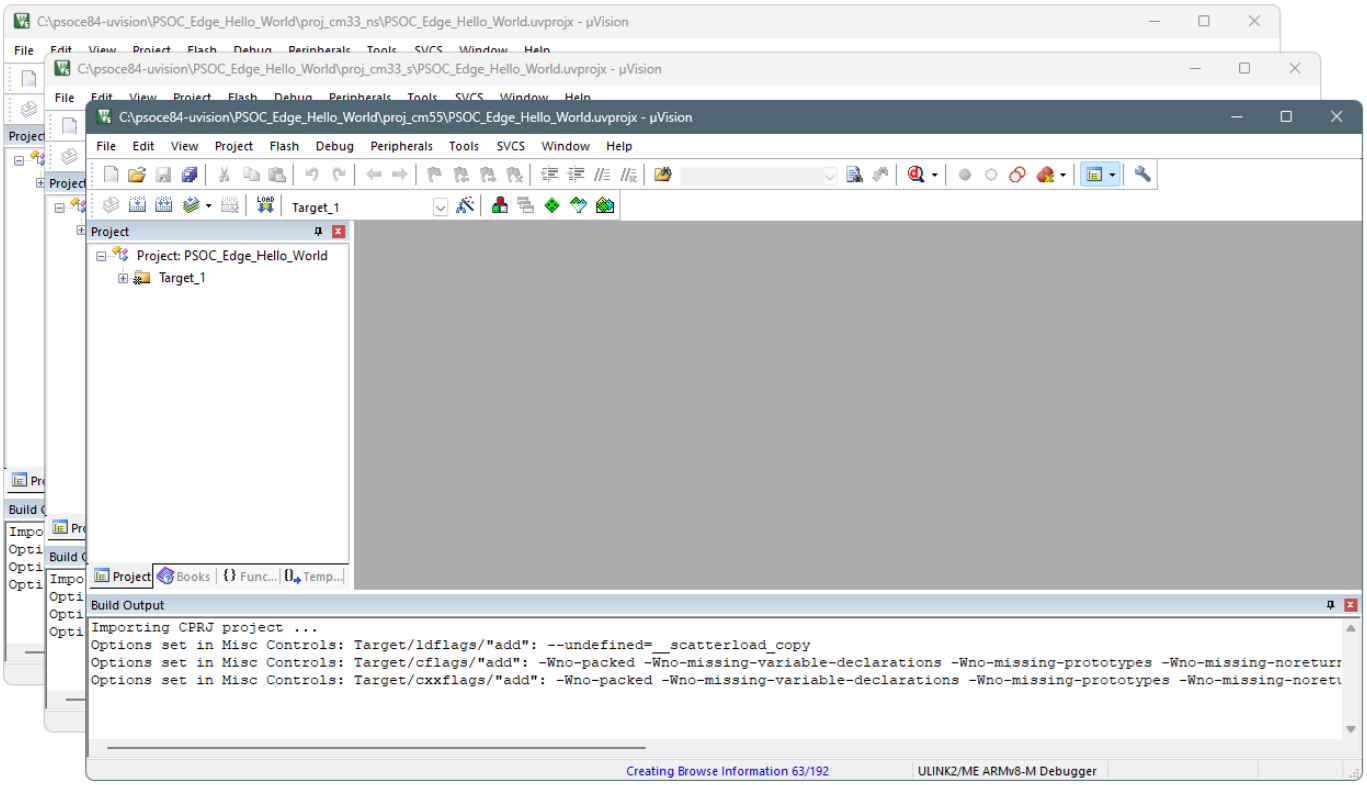
All other configuration steps are the same as those in Device with default policy.

Build the application. The post-build script will handle the signing process for you. After a successful build, you can program the device and start a debugging session.

For more details on device provisioning and hex file signing, refer Application Note AN240106 - Getting started with PSOC™ Control C3 security.

5 PSOC™ Edge E84 multi-core application

After creating all three projects as described in Create Keil µVision project(s). When complete, you should have three project instances of Keil µVision open like the following image. Then, follow the instructions in this section to configure, build, program, and debug the application.

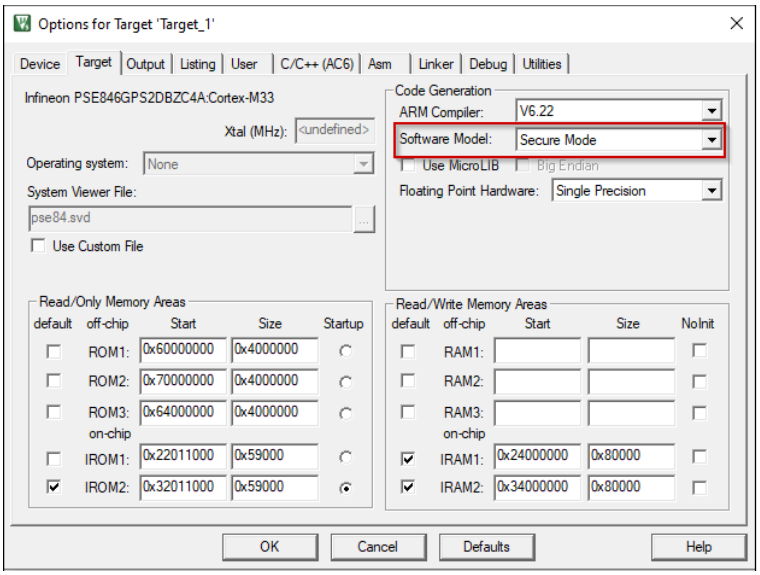


5.1 Configure and build multi-core projects

Once all projects are open, configure and build the secure CM33, non-secure CM33, and CM55 projects in sequential order. Because they are separate projects that depend on each other, the postbuild process will fail for the first two projects, but then will succeed on the third project.

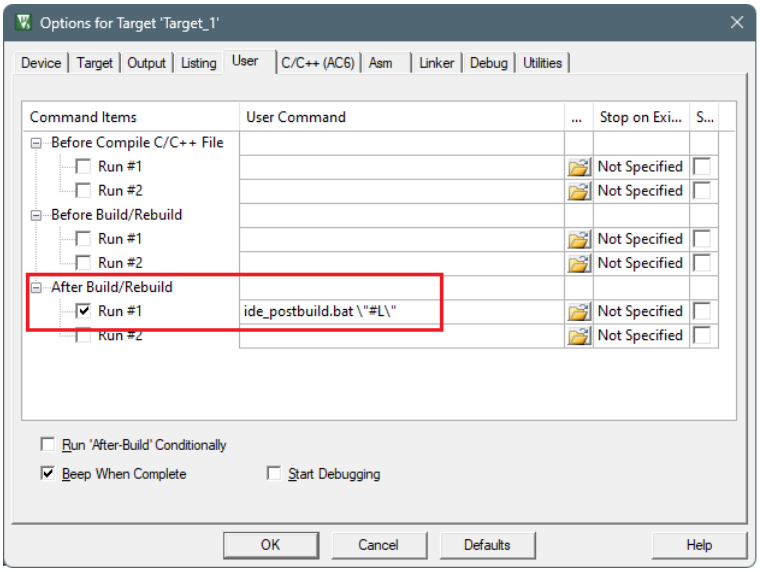
5.1.1 Secure CM33 project

1. Open the Options dialog. Go to the **Target** tab and select "Secure Mode" for **Software model**.



2. Switch to the **User** tab, enable the **Run #1** check box located under **After Build/Rebuild**. Then, paste the following command as single line (edit paths as needed).

```
ide_postbuild.bat \"#L\"
```



3. Select **Project > Build Target** to build the project.

This will generate an object file that needs to be referenced in the non-secure project in the next set of steps.

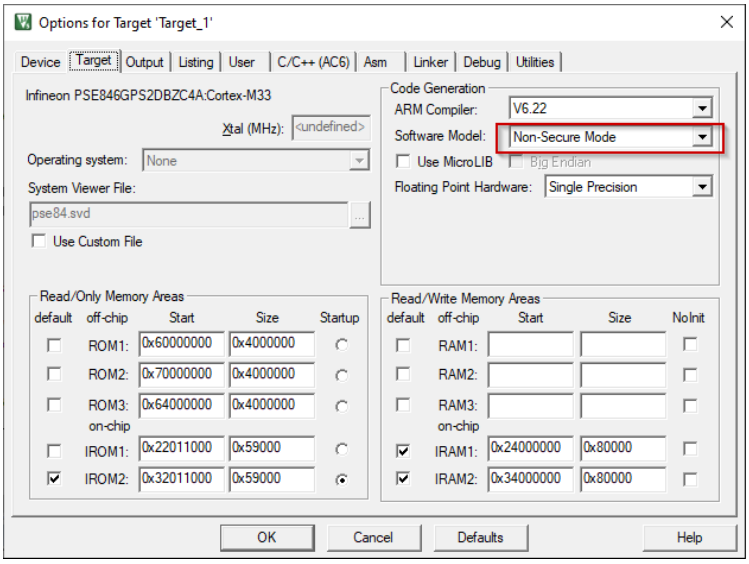
The postbuild will fail with a message like:

```
: E : ERROR : File not found: "C:\examples\edge-
uvision\PSOC_Edge_Hello_World\build\project_hex\proj_cm33_ns.hex". Check the log for details
```

Proceed to the next step to configure the cm33_ns project.

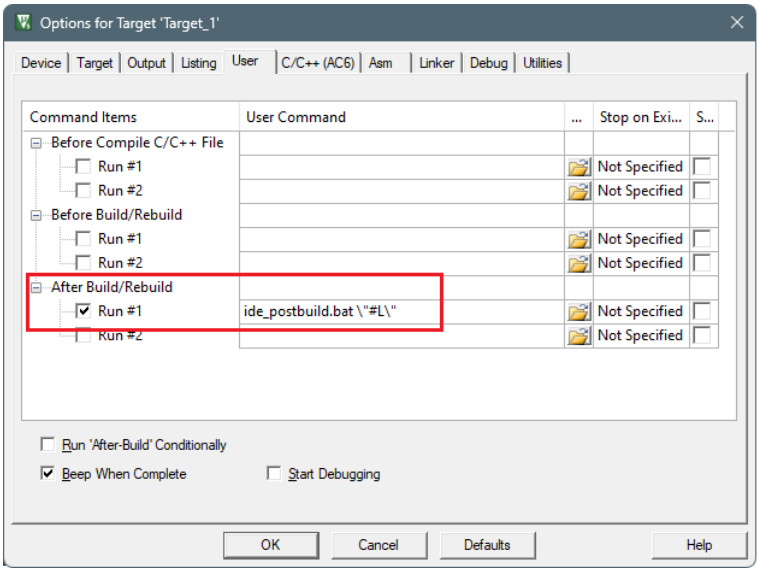
5.1.2 Non-secure CM33 project

1. Open the Options dialog. Go to the **Target** tab and select "Non-Secure Mode" for **Software model**.



2. Switch to the **User** tab and enable the **Run #1** check box located under **After Build/Rebuild**. Then, paste the following command as single line (edit paths as needed).

```
ide_postbuild.bat \"#L\"
```



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

4. Select **Project > Build Target** to build the project.

The postbuild will fail with a message like:

```
: E : ERROR : File not found: "C:\examples\edge-
uvision\PSOC_Edge_Hello_World\build\project_hex\proj_cm55.hex". Check the log for details
```

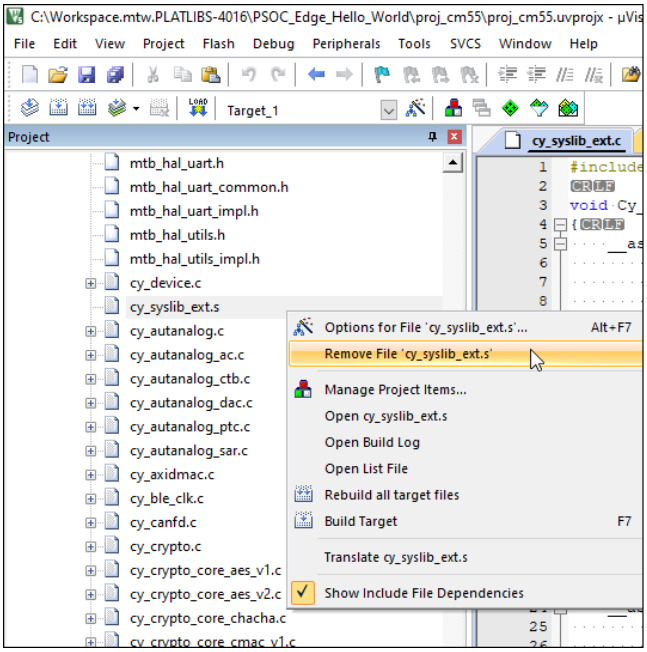
Proceed to the next step to configure the cm55 project.

5.1.3 CM55 project

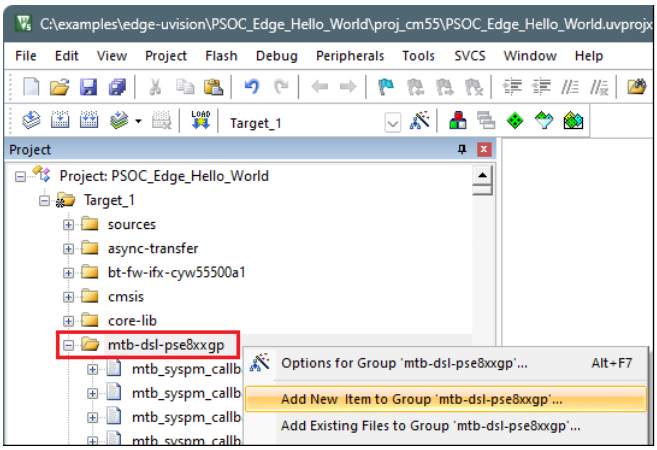
The default CM55 project includes a *cy_syslib_ext.s* that is not compatible with µVision, and the file needs to replaced with a .c file

1. Select the project window and locate the file *cy_syslib_ext.s* under the **Target_1 > mtb-dsl-pse8xxgp** in the project tree. Right-click on it and select **Remove File 'cy_syslib_ext.s'**.

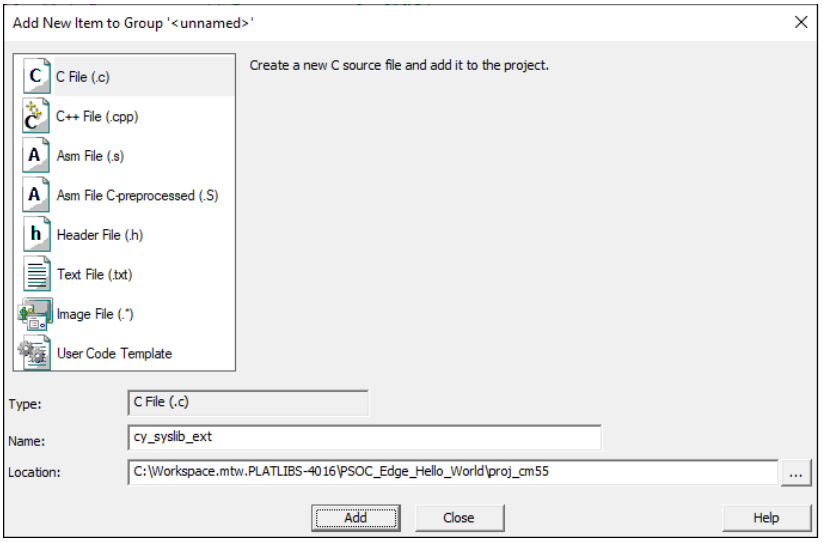
   ***Note**: In some older versions of µVision, there is only one folder named '<unnamed>'.*



2. Right click on the **mtb-dsl-pse8xxgp** group and select **Add New Item to Group '[name]' ...**



3. Then, choose a new C source file, specify the name "cy_syslib_ext", and click the **Add** button.



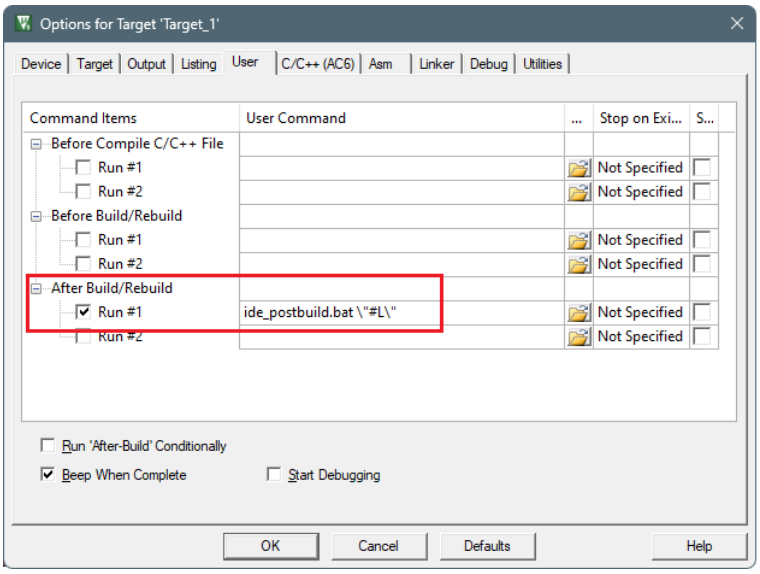
4. Copy the following code and paste it into the newly created file named "cy_syslib_ext.c".

```
#include <stdint.h>

void Cy_SysLib_DelayCycles(uint32_t cycles)
{
    __asm volatile (
        "adds %0, %0, #2 \n"           // Add 2 to cycles
        "lsrs %0, %0, #2 \n"           // Logical Shift Right by 2 (divide by 4)
        "beq Cy_DelayCycles_done \n"   // Branch to done if cycles is zero

        "Cy_DelayCycles_loop: \n"      // Label for loop start
        "adds %0, %0, #1 \n"           // Increment cycles
        "subs %0, %0, #2 \n"           // Subtract 2 from cycles
        "bne Cy_DelayCycles_loop \n"   // Branch to loop if cycles is not zero

        "Cy_DelayCycles_done: \n"      // Label for loop end
        : "+r" (cycles)                // Output operand
    );

}

uint32_t Cy_SysLib_EnterCriticalSection(void)
{
    uint32_t primask;
    __asm volatile(
        "mrs %0, primask\n"    // Save and return interrupt state
        "cpsid i"              // Disable interrupts
        : "=r" (primask)       // Output to 'primask'
        :
        : "memory"
    );
    return primask;
}

void Cy_SysLib_ExitCriticalSection(uint32_t savedIntrStatus)
{
    __asm volatile(
        "msr primask, %0"      // Restore interrupt state
        :
        : "r" (savedIntrStatus) // Input from 'savedIntrStatus'
        : "memory"
    );
}
```

5. Save the file.

6. Open the Options dialog. Under the **User** tab, enable the **Run #1** check box located under **After Build/ Rebuild**. Then, paste the following command as single line.

```
ide_postbuild.bat \"#L\"
```



7. Click **OK** to close the Options dialog.
8. Select **Project > Build Target** to build the project and execute post-build commands.

This time, the postbuild will succeed, and you'll see messages similar to the following:

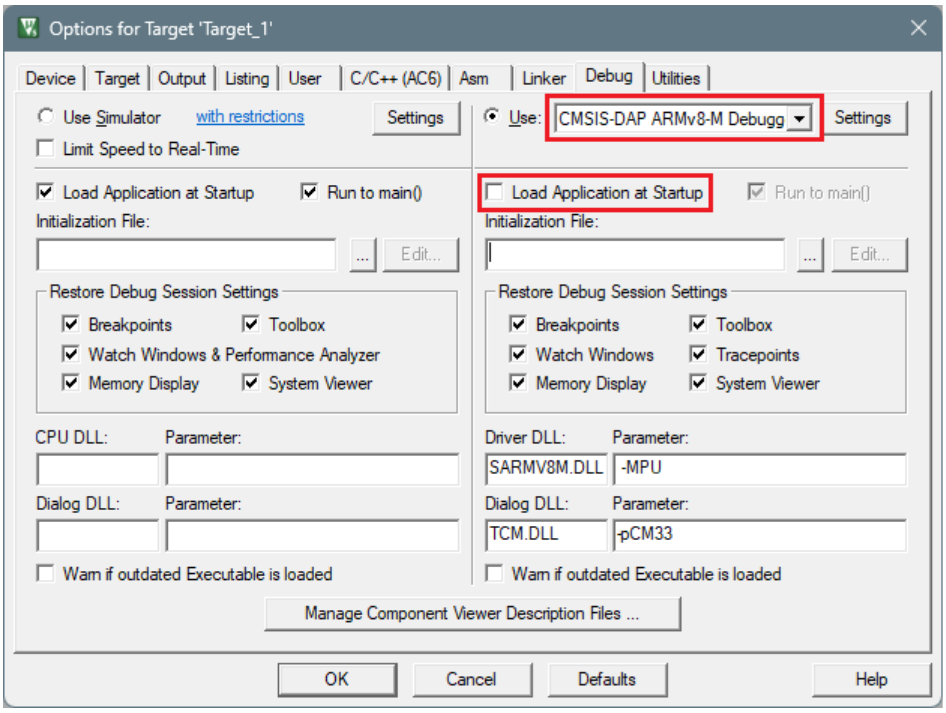
```
: E : INFO  : metadata_proj_cm33_s: command "sign" validation succeeded
: E : INFO  : Image saved to 'C:\examples\edge-
uvision\PSOC_Edge_Hello_World\build\project_hex\proj_cm33_s_signed.hex'
: E : INFO  : metadata_proj_cm33_s: command "sign" succeeded
: E : INFO  : relocate_proj_cm33_ns: command "hex-relocate" validation succeeded
: E : INFO  : Relocating segment 0x08340400-0x08344100 to 0x60340400-0x60344100
: E : INFO  : Saved file to '../build/project_hex/proj_cm33_ns_shifted.hex'
: E : INFO  : relocate_proj_cm33_ns: command "hex-relocate" succeeded
: E : INFO  : merge: command "merge" validation succeeded
: E : INFO  : merge: command "merge" succeeded
Searching installed tools in progress...
Searching installed tools complete
C:\examples\edge-uvision\PSOC_Edge_Hello_World\proj_cm55>ENDLOCAL
".\PSOC_Edge_Hello_World_Objects\PSOC_Edge_Hello_World.axf" - 0 Error(s), 39 Warning(s).
Build Time Elapsed:  00:01:02
```

5.2 Debugger configuration

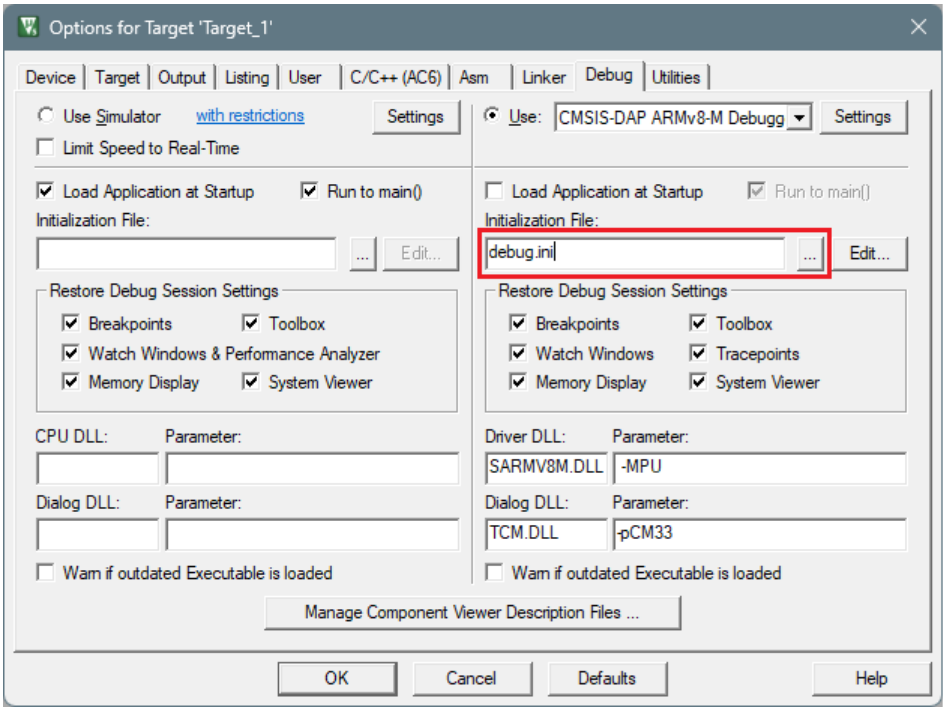
Next, configure projects to launch multi-core debugging.

5.2.1 Secure CM33 project

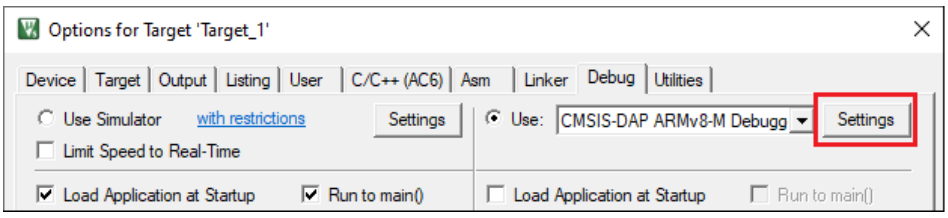
1. Open the Options dialog. Select the **Debug** tab and select the applicable debug probe (CMSIS-DAP or J-Link). Make sure that the **Load Application at Startup** check box is not selected.



2. Select the *debug.ini* file in the **Initialization File** field.

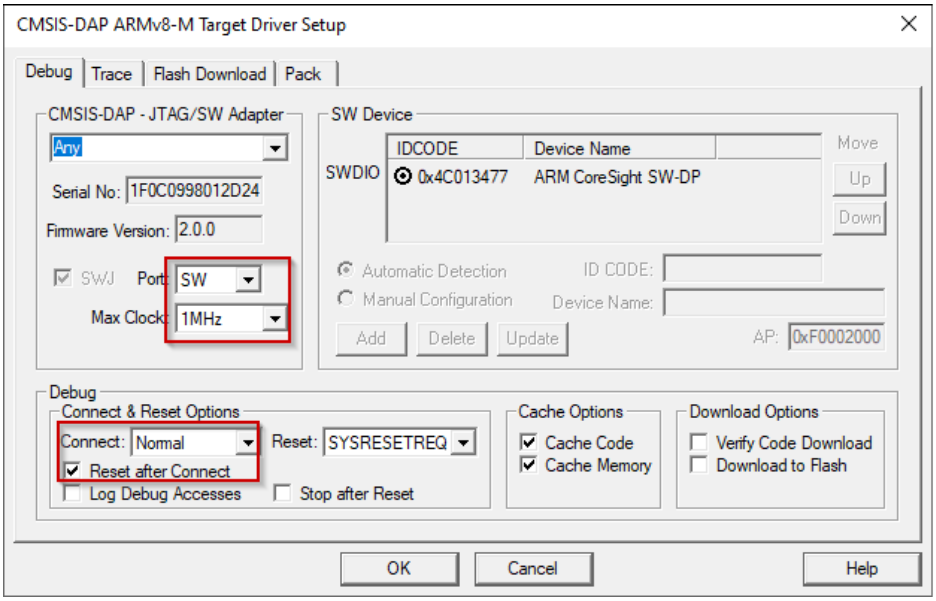


3. Click the **Settings** button to configure the target driver.



4. Select the **Debug** tab, and choose applicable connection options, the configuration settings are different for CMSIS-DAP and J-Link. Refer to the following sections for the applicable options:

CMSIS-DAP/ULINK2 Target Driver Setup

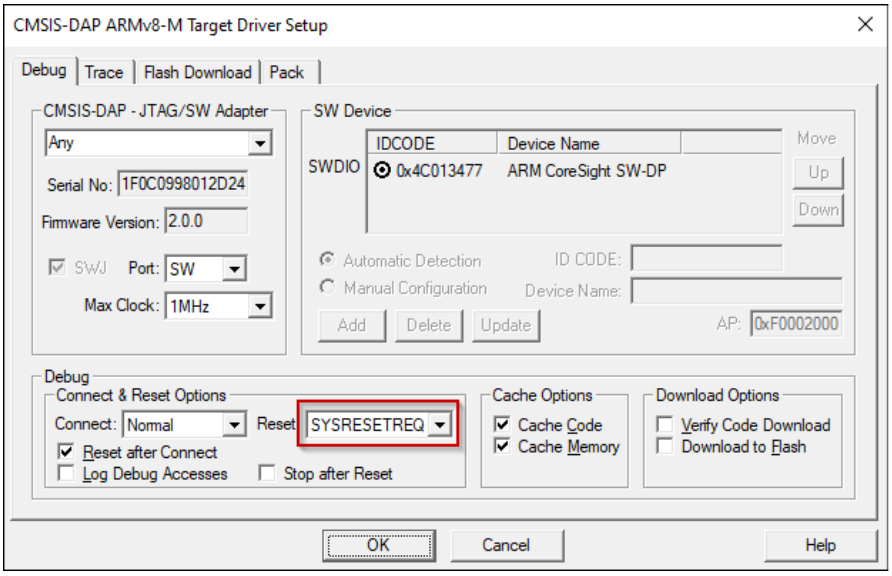


Use the following options:

- **Port**: SW
- **Max Clock**: 1 MHz
- **Connect**: Normal
- **Reset after Connect**: selected

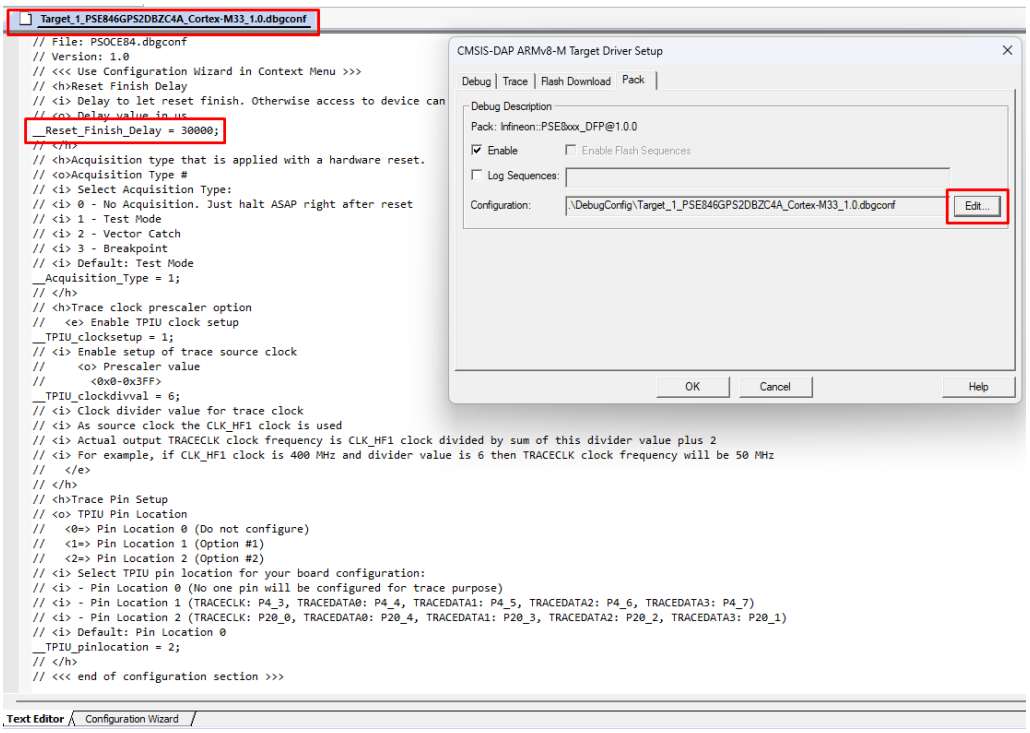
Select desired **Reset** type. For PSOC™ Edge devices, there are two main reset types available: hardware reset using XRES and system reset by SYSRESETREQ.

- For SYSRESETREQ the Vector Catch method is applied. The default reset type is SYSRESETREQ.



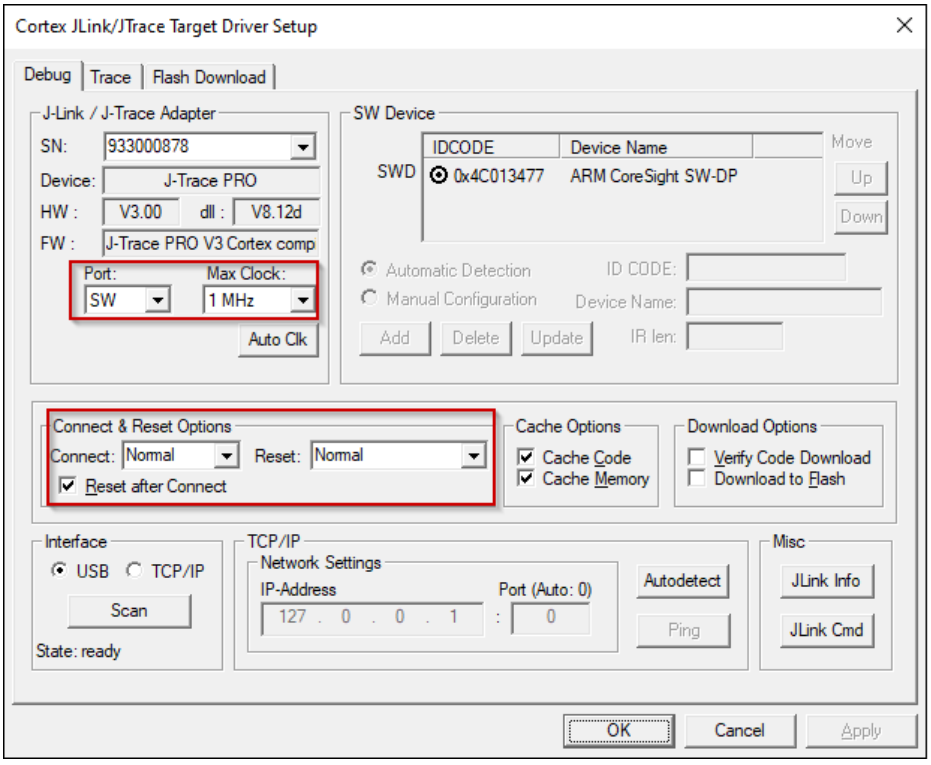
- For XRES additional follow-up acquisition methods are available, including Test Mode acquisition, Vector Catch, and Break Point acquisition (an alternative acquire).

The method to use is determined by the ***__Acquisition_Type*** debug variable. The debug variables are accessible for editing at tab Pack clicking **Edit** button.



By default, the Test Mode acquisition is applied after XRES.

J-Link Target Driver Setup

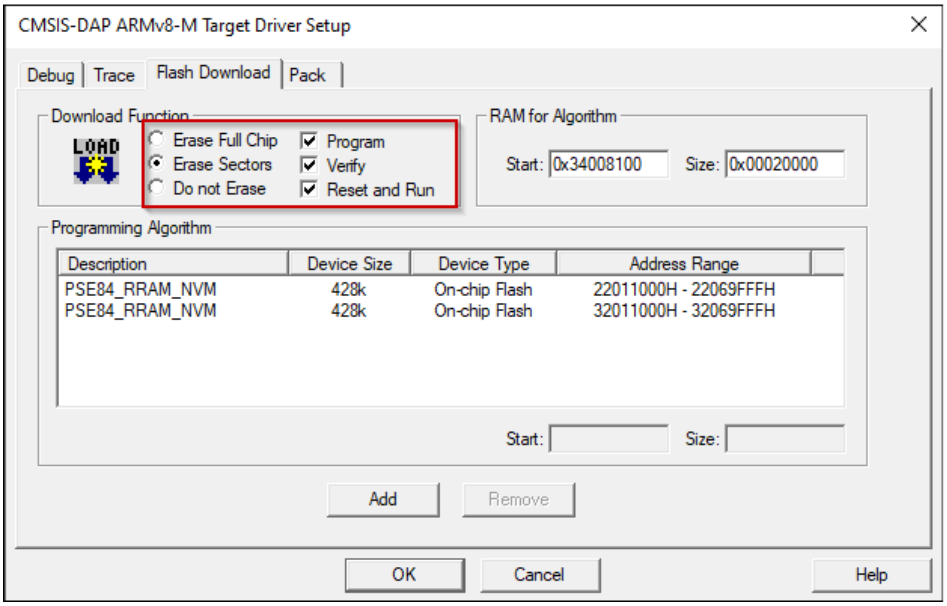


Use the following options:

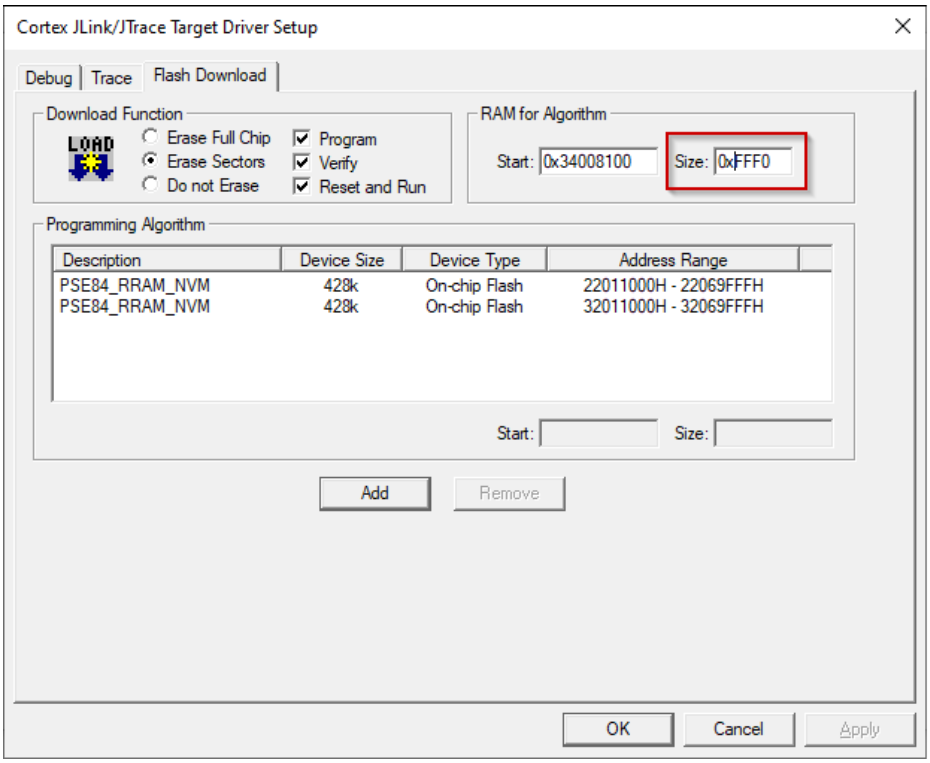
- **Port**: SW
- **Max Clock**: 1 MHz

- **Connect**: Normal
- **Reset**: Normal
- **Reset after Connect**: selected

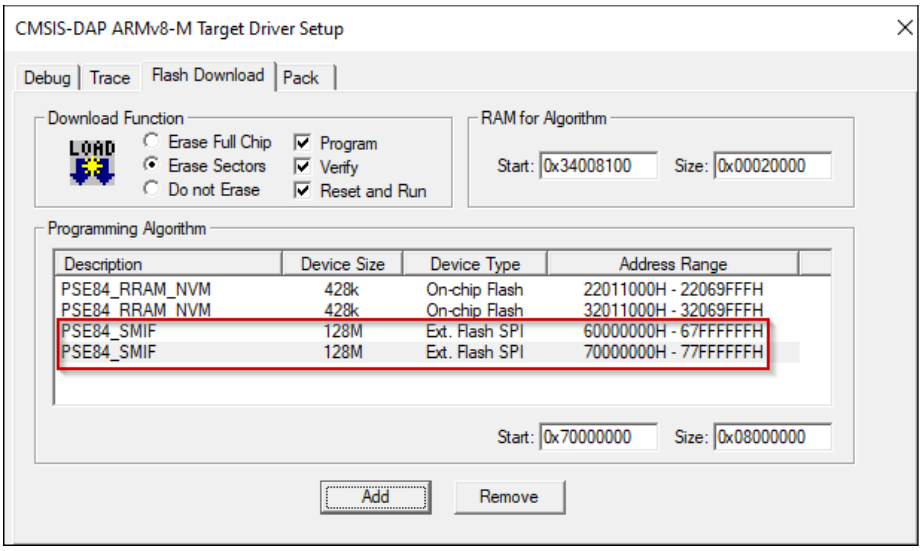
5. For either driver, switch to the **Flash Download** tab and select **Erase Sectors** radio button, as well as **Program**, **Verify**, and **Reset and Run** check boxes.



6. For J-Link, ensure the RAM size for the algorithm is set to 0xFFF0.



7. For either driver, if any application (secure CM33, non-secure CM33, or CM55) should be programmed to external memory, two additional instances of flashloaders with name PSE84_SMIF need to be added using the **Add** button.

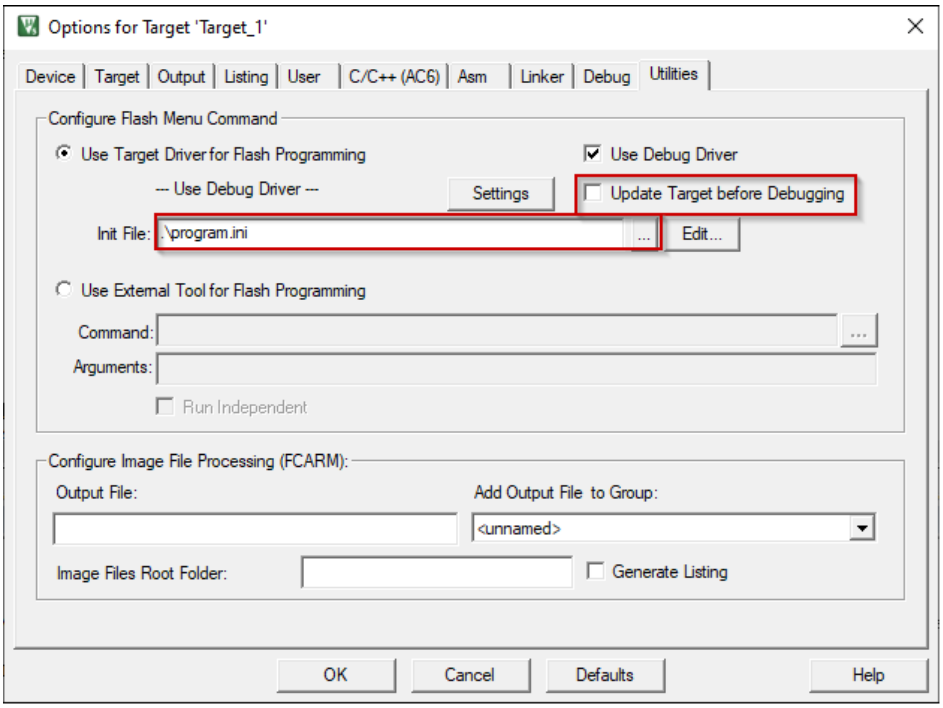


Make sure the second instance handle start address is 0x70000000.

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

9. Close the Options dialog.

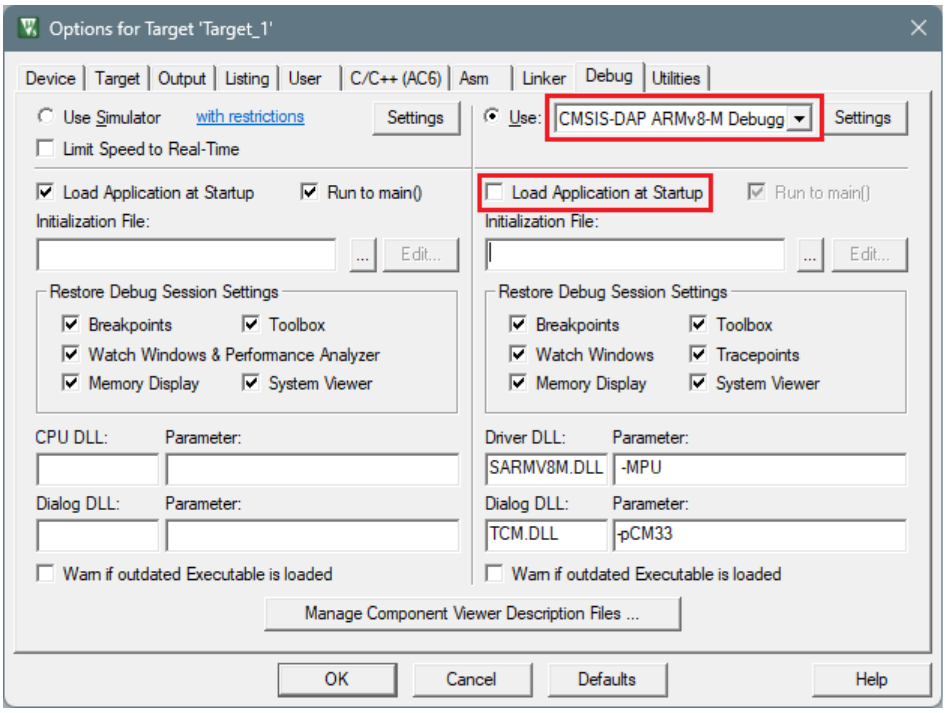
10. Open the Options dialog again and select the **Utilities** tab. Deselect the check box **Update Target before Debugging** and select the *program.ini* file in the **Init File** field.



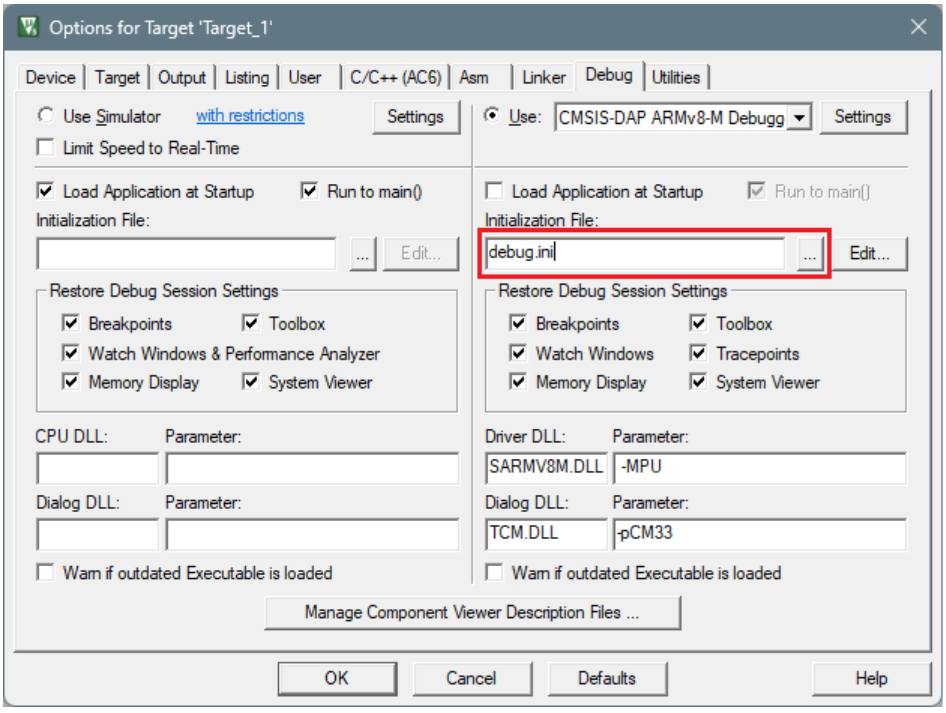
11. Close the Options dialog and save the project.

5.2.2 Non-Secure CM33 project

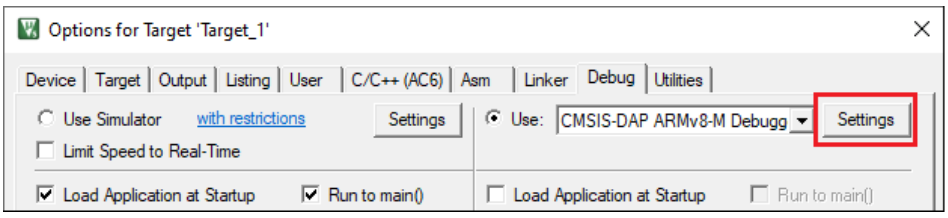
1. Open the Options dialog. Select the **Debug** tab and select the applicable debug probe (CMSIS-DAP or J-Link). Make sure that the **Load Application at Startup** check box is not selected.



2. Select the *debug.ini* file in the **Initialization File** field.

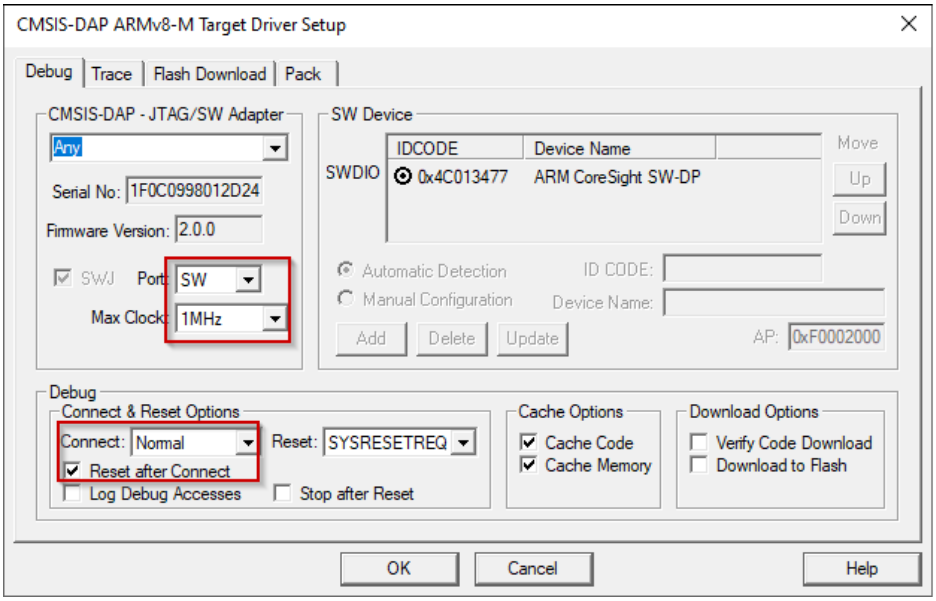


3. Click the **Settings** button to configure the target driver.



4. Select the **Debug** tab, and choose applicable connection options, the configuration settings are different for CMSIS-DAP and J-Link. Refer to the following sections for the applicable options:

CMSIS-DAP/ULINK2 Target Driver Setup

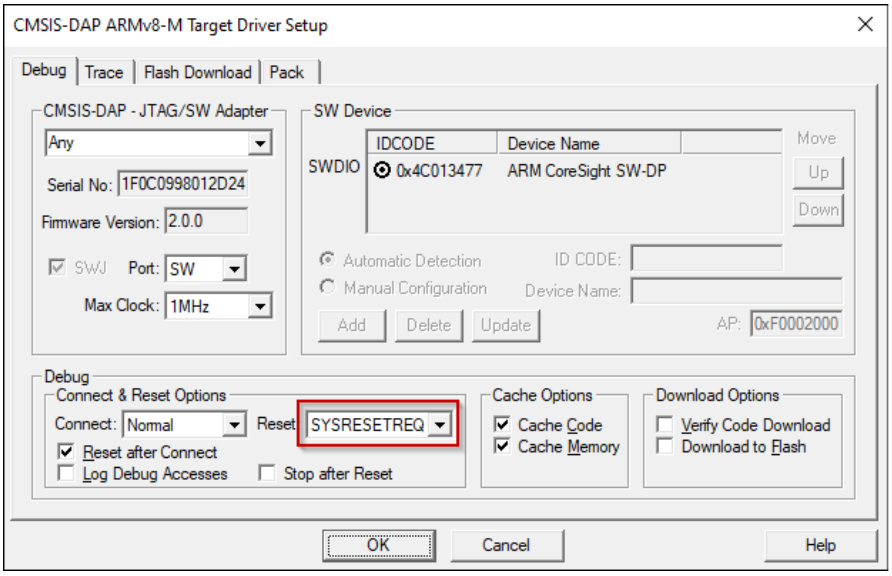


Use the following options:

- **Port**: SW
- **Max Clock**: 1 MHz
- **Connect**: Normal
- **Reset after Connect**: selected

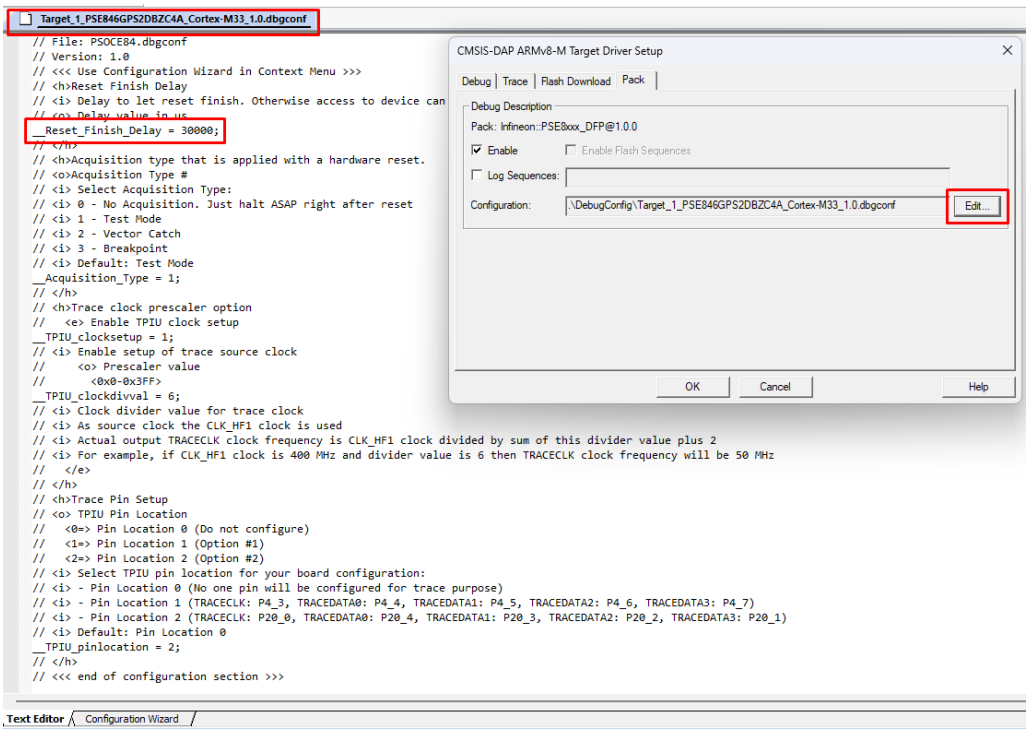
Select desired **Reset** type. For PSOC™ Edge devices, there are two main reset types available: hardware reset using XRES and system reset by SYSRESETREQ.

- For SYSRESETREQ the Vector Catch method is applied. The default reset type is SYSRESETREQ.



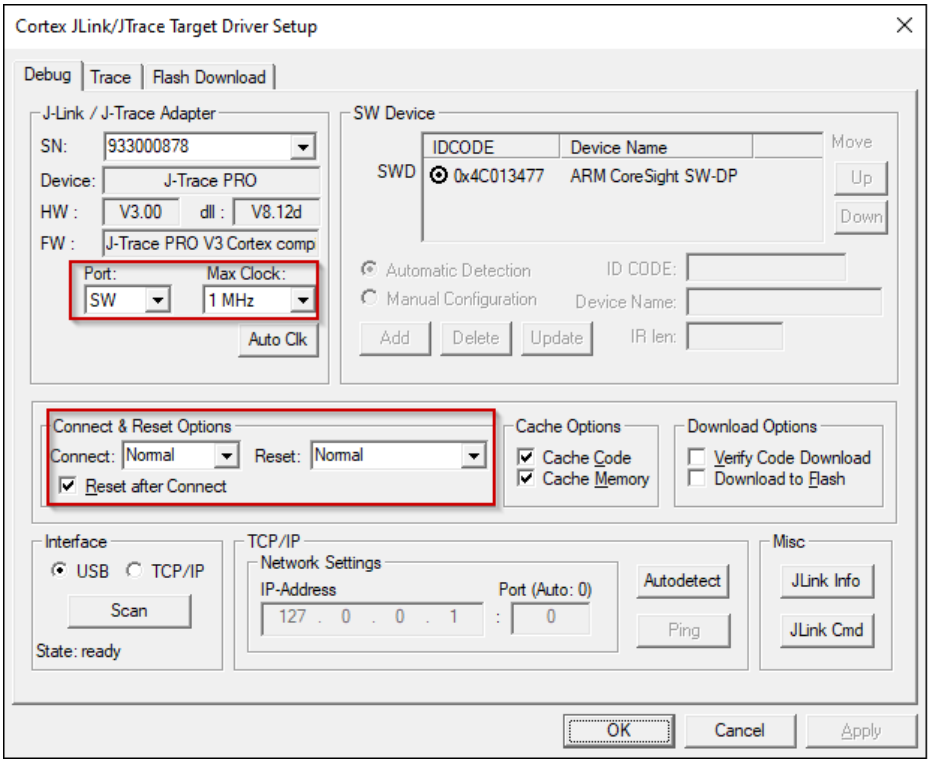
- For XRES additional follow-up acquisition methods are available, including Test Mode acquisition, Vector Catch, and Break Point acquisition (an alternative acquire).

The method to use is determined by the ***__Acquisition_Type*** debug variable. The debug variables are accessible for editing at tab Pack clicking **Edit** button.



By default, the Test Mode acquisition is applied after XRES.

J-Link Target Driver Setup

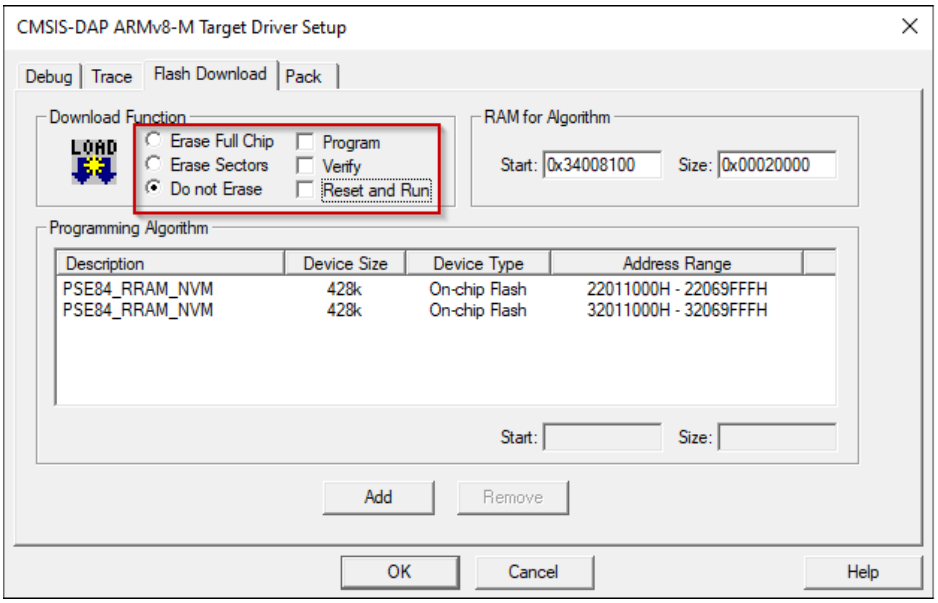


Use the following options:

- **Port**: SW
- **Max Clock**: 1 MHz

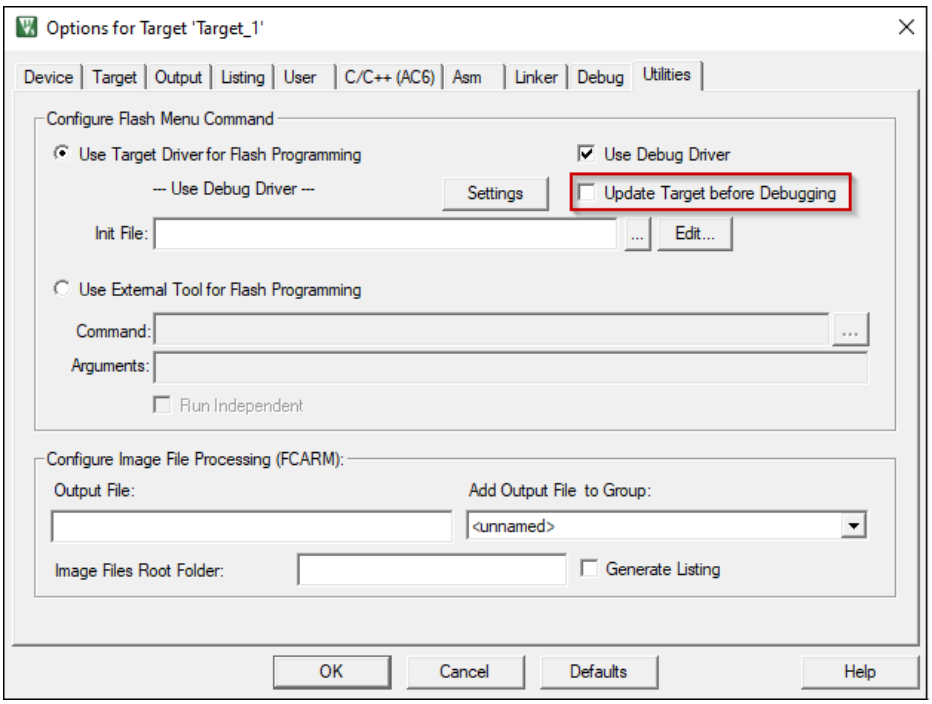
- **Connect**: Normal
- **Reset**: Normal
- **Reset after Connect**: selected

5. For either driver, switch to the **Flash Download** tab and select **Do not Erase Sectors** radio button, and deselect **Program**, **Verify**, and **Reset and Run** check boxes.



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

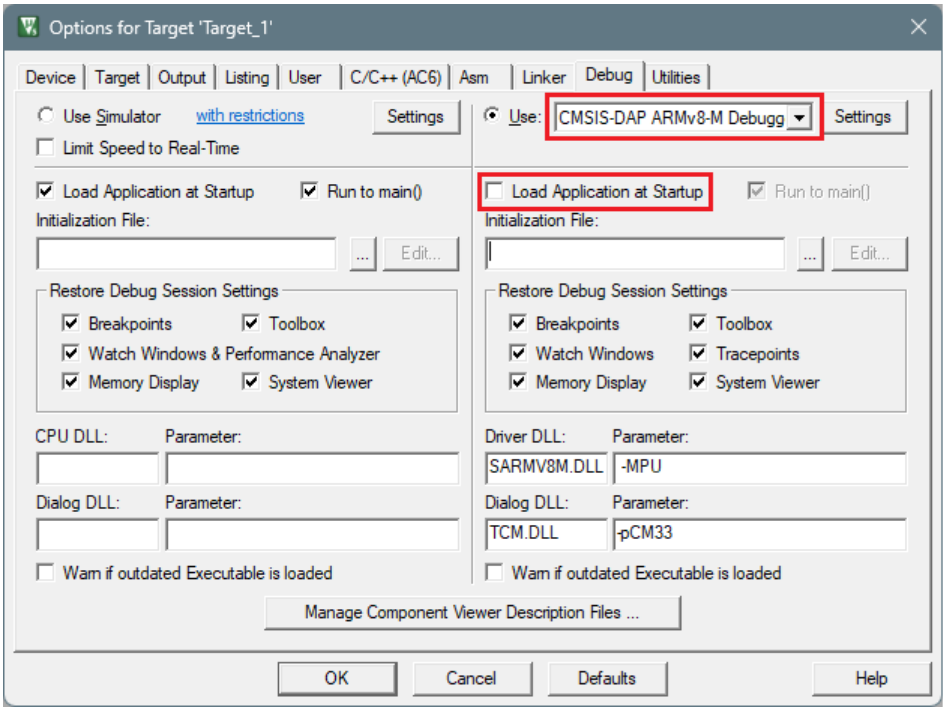
7. Switch to the **Utilities** tab. Deselect the check box **Update Target before Debugging**.



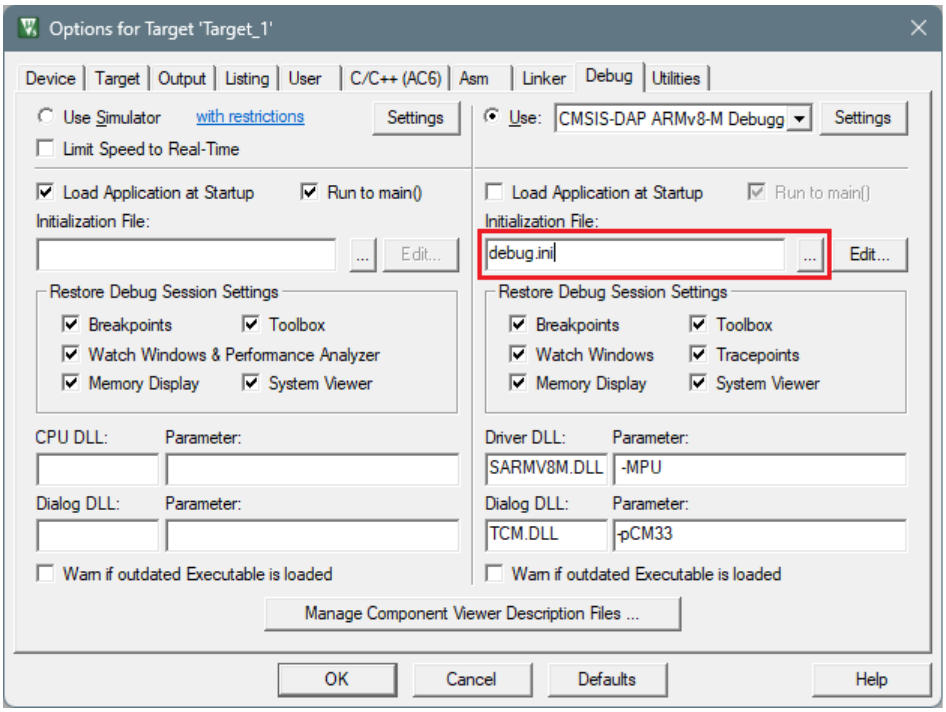
8. Close the Options dialog and save the project.

5.2.3 CM55 project

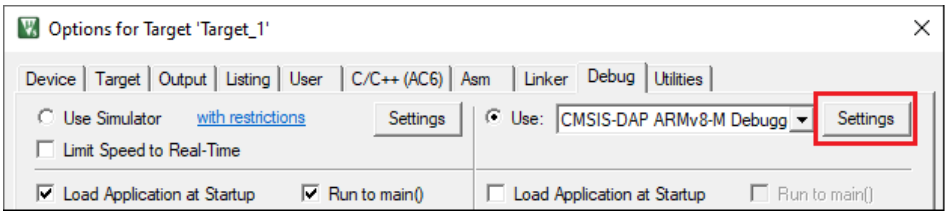
1. Open the Options dialog. Select the **Debug** tab and select the applicable debug probe (CMSIS-DAP or J-Link). Make sure that the **Load Application at Startup** check box is not selected.



2. Select the *debug.ini* file in the **Initialization File** field.

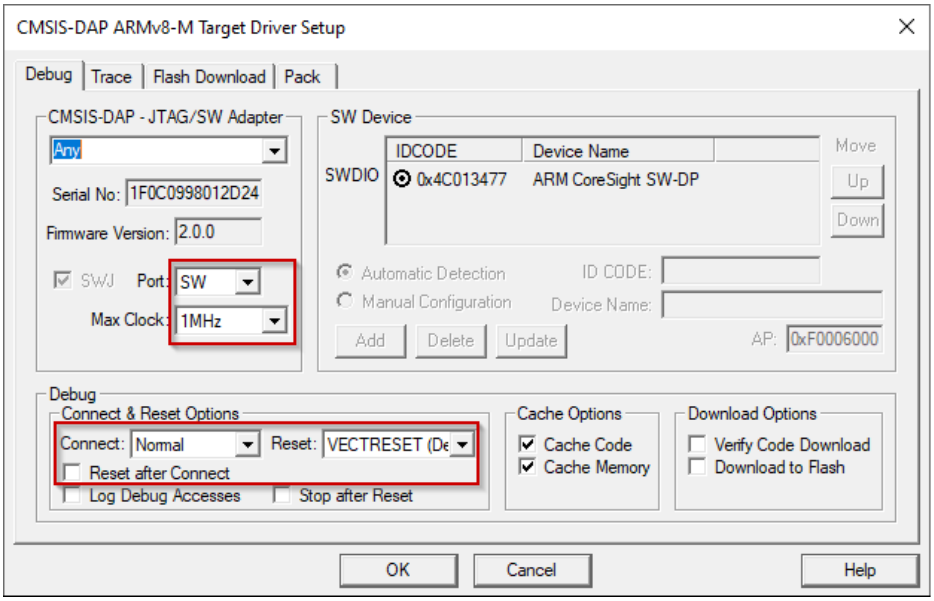


3. Click the **Settings** button to configure the target driver.



4. Select the **Debug** tab, and choose applicable connection options, the configuration settings are different for CMSIS-DAP and J-Link. Refer to the following sections for the applicable options:

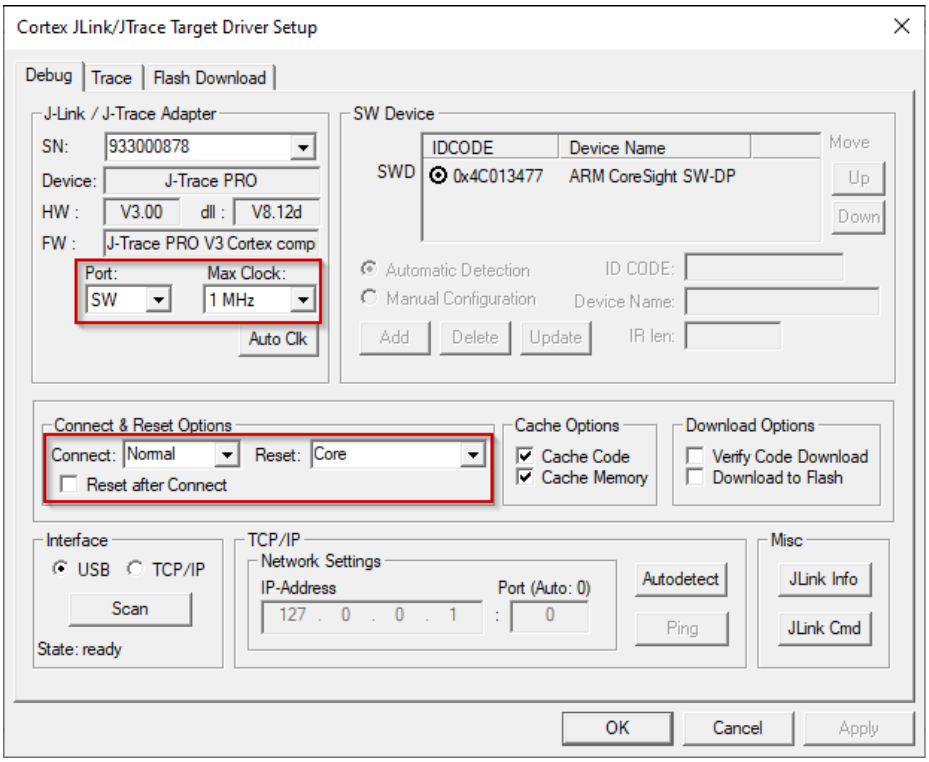
CMSIS-DAP/ULINK2 Target Driver Setup



Use the following options:

- **Port**: SW
- **Max Clock**: 1 MHz
- **Connect**: Normal
- **Reset**: VECTRESET
- **Reset after Connect**: Not selected

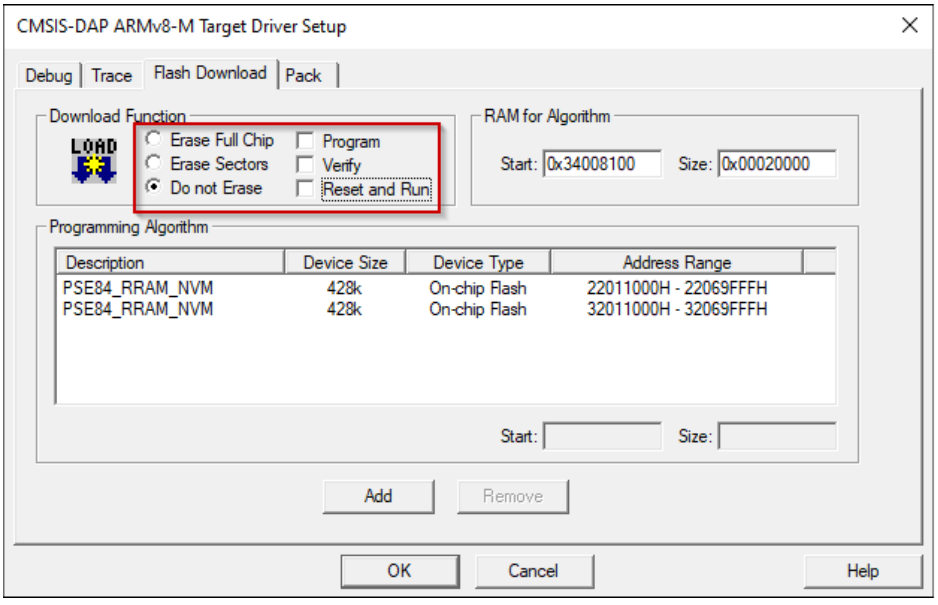
J-Link Target Driver Setup



Use the following options:

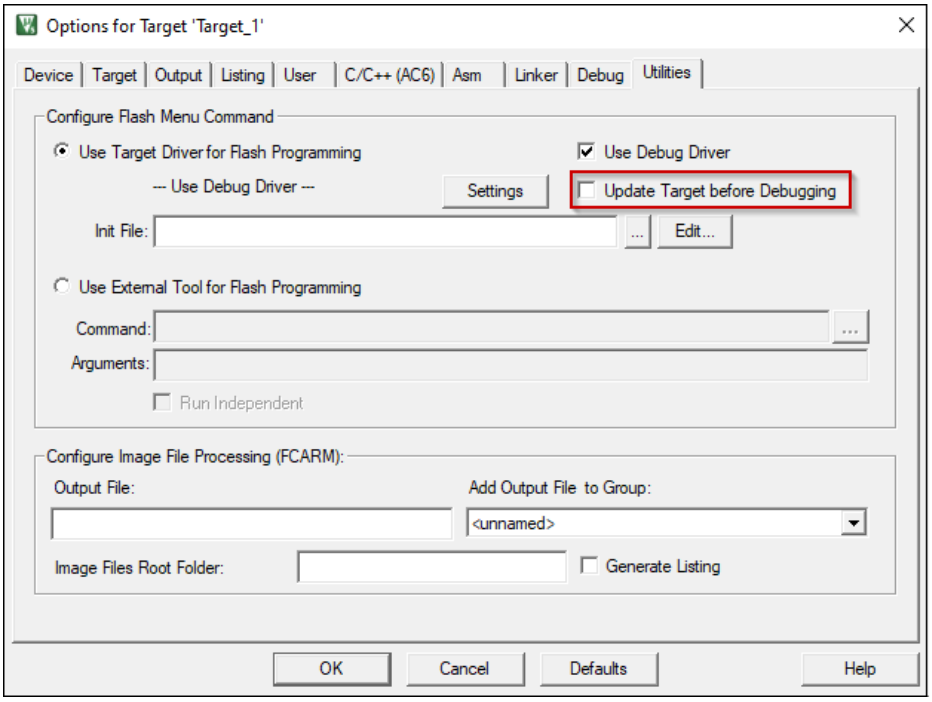
- **Port**: SW
- **Max Clock**: 1 MHz
- **Connect**: Normal
- **Reset**: Core
- **Reset after Connect**: Not selected

5. For either driver, switch to the **Flash Download** tab and select **Do not Erase Sectors** radio button, and deselect **Program**, **Verify**, and **Reset and Run** check boxes.



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

7. Switch to the **Utilities** tab. Deselect the check box **Update Target before Debugging**.



8. Close the Options dialog and save the project.

5.2.4 Secure debug configuration (optional)

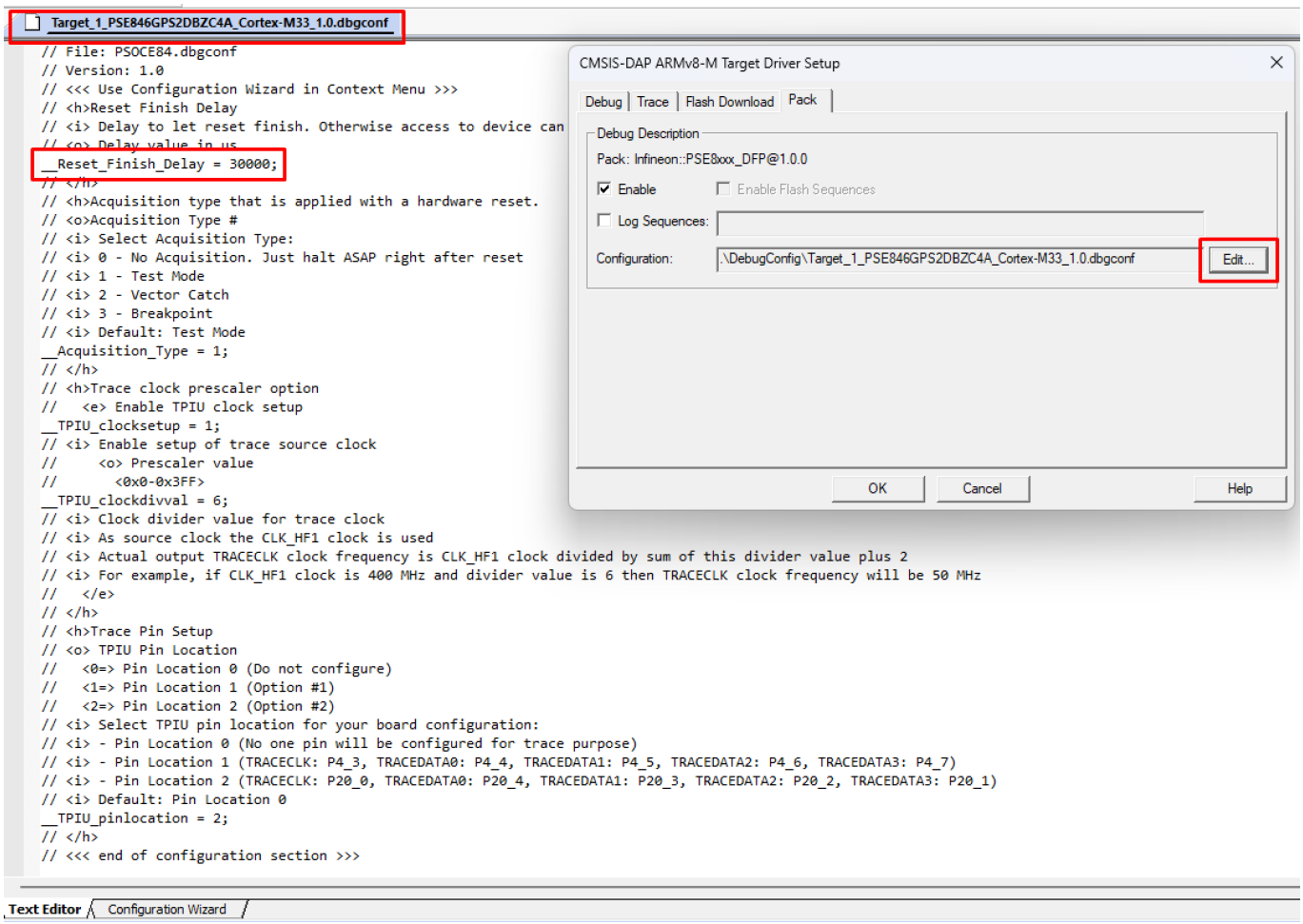
The BootROM on the PSOC™ Edge can temporarily disable access to the core's AP. This behavior is determined by the security policy during device provisioning. However, access can be re-enabled using a debug certificate. If a debugger identifies that the AP is disabled, it will upload and verify the certificate found in the "packets/debug_token.bin" file within the current project folder. It's essential to note that neither the name of the certificate file nor its location cannot be changed.

5.3 Recommended Values for __Reset_Finish_Delay

Proper device operation may require adjusting the `__Reset_Finish_Delay` parameter, depending on the device's configuration. The following show the recommended values for this parameter when using KitProg3/MiniProg4 or ULINK2 debuggers/programmers.

- KitProg3/MiniProg4 – 20 000 - 100 000
- ULINK2 – 10 000 - 100 000

For example:

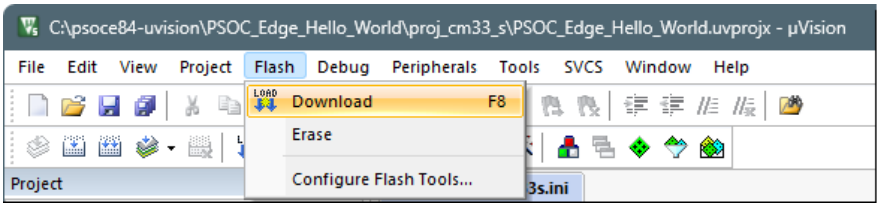


5.4 Target programming

After preparing the combined image and setting up the debugger connection, the next step is to program the target. To accomplish this, follow these steps:

1. Select the µVision instance designated for the secure CM33 application.

2. On the main menu, select **Flash > Download**.



5.5 Launch multi-core debug session

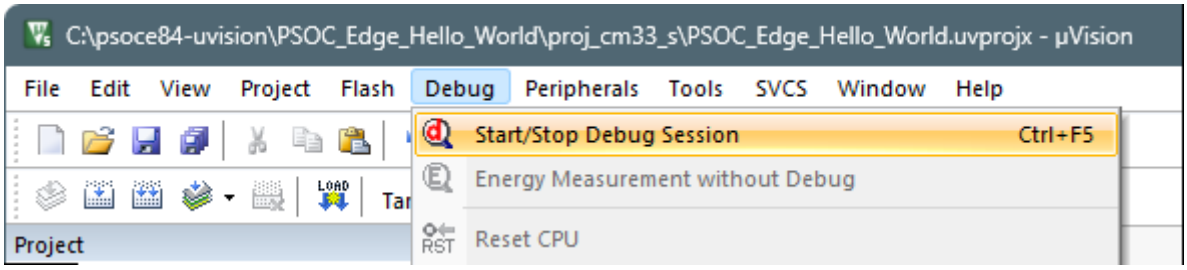
Since two instances of µVision cannot debug the same core simultaneously, it is not possible to debug a secure CM33 application and a non-secure CM33 application at the same time. The following multi-core debug combinations are possible:

- Debugging a secure CM33 project alongside a CM55 project.
- Debugging a non-secure CM33 project alongside a CM55 project.

5 PSOC™ Edge E84 multi-core application

To initiate a multi-core debug session, you must open one project for the CM33 core (secure or non-scure) and another for the CM55 core in separate instances of the IDE. Here are the steps:

1. Select the µVision IDE instance with the project for the CM33 core and start debugging by selecting **Debug > Start/Stop Debug Session**.



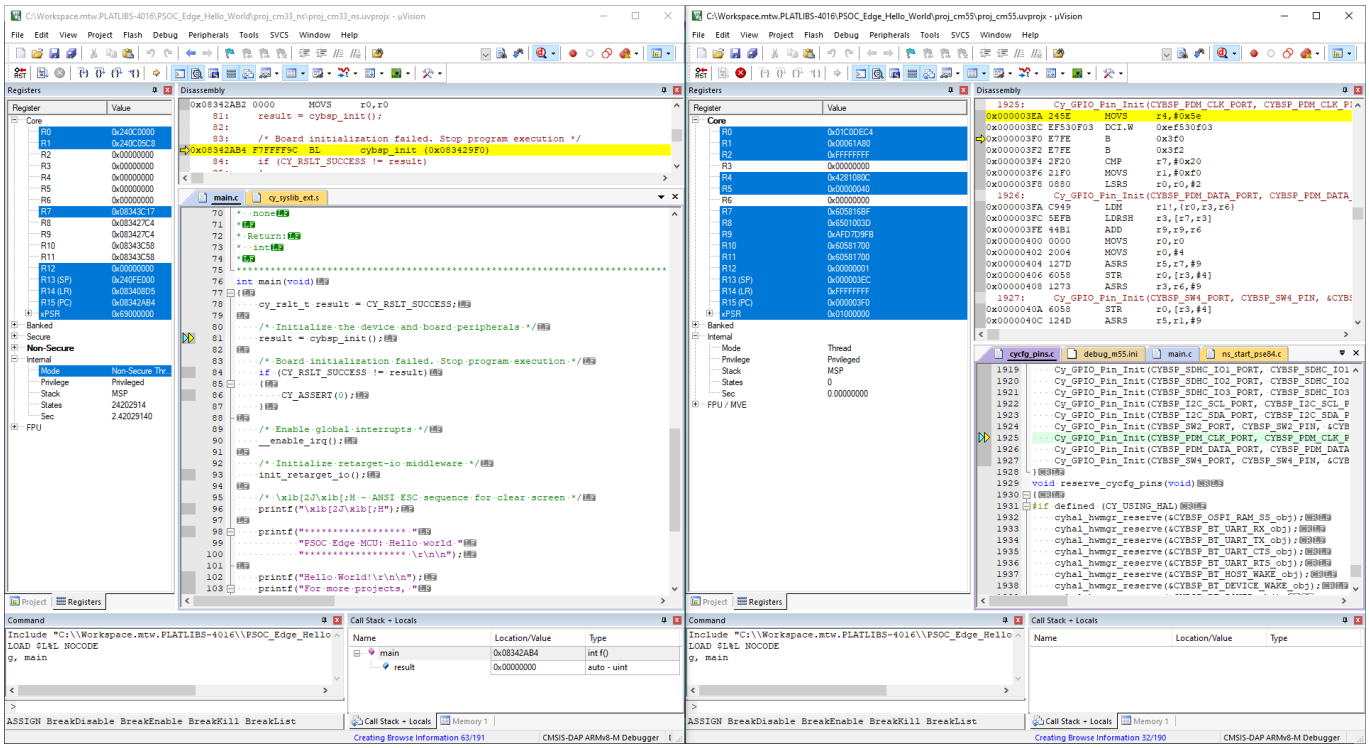
This will reset the target and halt execution at the beginning of the `main()` function of the CM33 project.

2. Select the µVision IDE instance with the project for the CM55 core and repeat the debugging process.

This will attach to the running CM55 core, which will be in boot mode until the CM33 project starts executing.

***Note**: Ensure all projects are built and programmed before launching a debug session.*

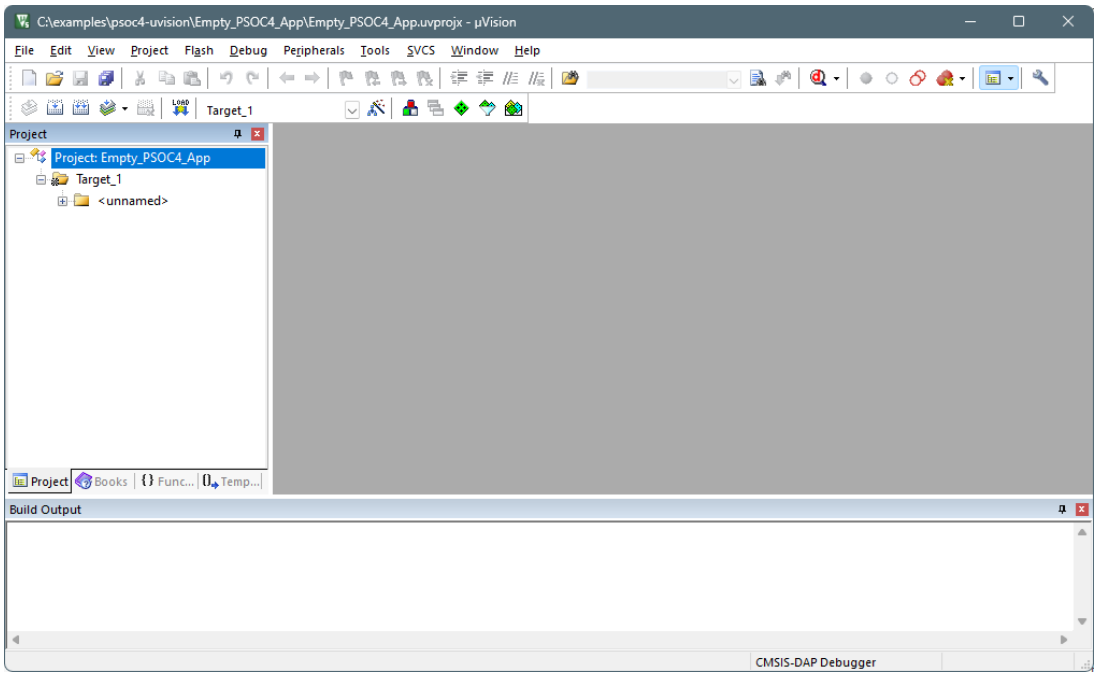
The projects will be similar to these images:



The left side of the screen shows a µVision IDE instance attached to the CM33 core. The right side shows the CM55 core has not started yet. Once the `Cy_SysEnableCM55()` function on the CM33 core has been executed, the CM55 will start executing its application. You can step through the code by switching back and forth between the two µVision IDE instances.

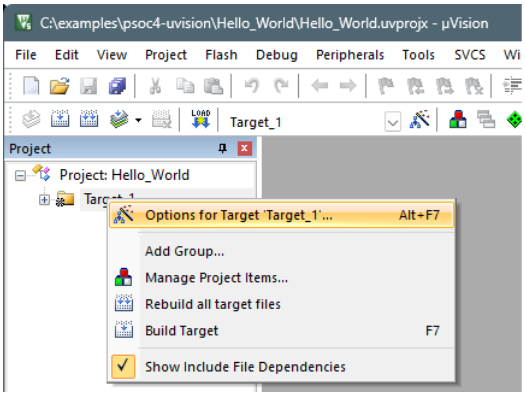
6 PSOC™ 4 and PSOC™ 6 single-core application

Follow instructions in Create/export application for Keil µVision. When complete, your project should look similar to the following. Then, use the instructions in this section to configure, build, and debug your application.



6.1 Configure and build the application

1. Right-click on the *Target_1* directory in the µVision Project view and select **Options for Target 'Target_1'** **...**

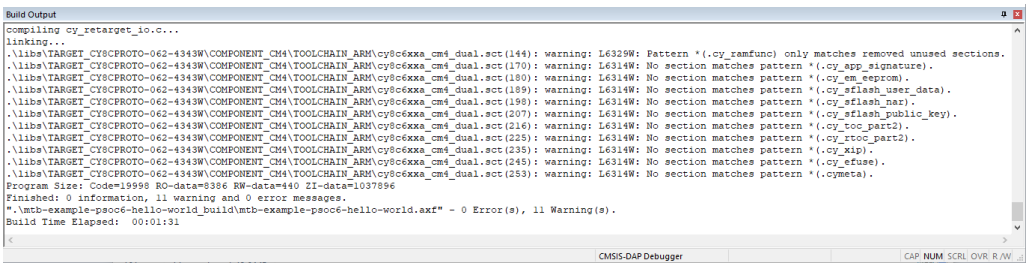


2. Select the **Debug** tab and then select the appropriate debugger probe. See Miscellaneous notes for details to configure the probe.

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

   ***Note**: For applications using the PSOC™ 64 secure single-core application or AIROC™ CYW20829 single-core application, skip the next step. Instead, perform the steps outlined in the applicable section.*

4. Select **Project > Build target**.



6.2 Programming/Debugging

Refer to sections for setting up the applicable debugger probe under Miscellaneous notes. Then:

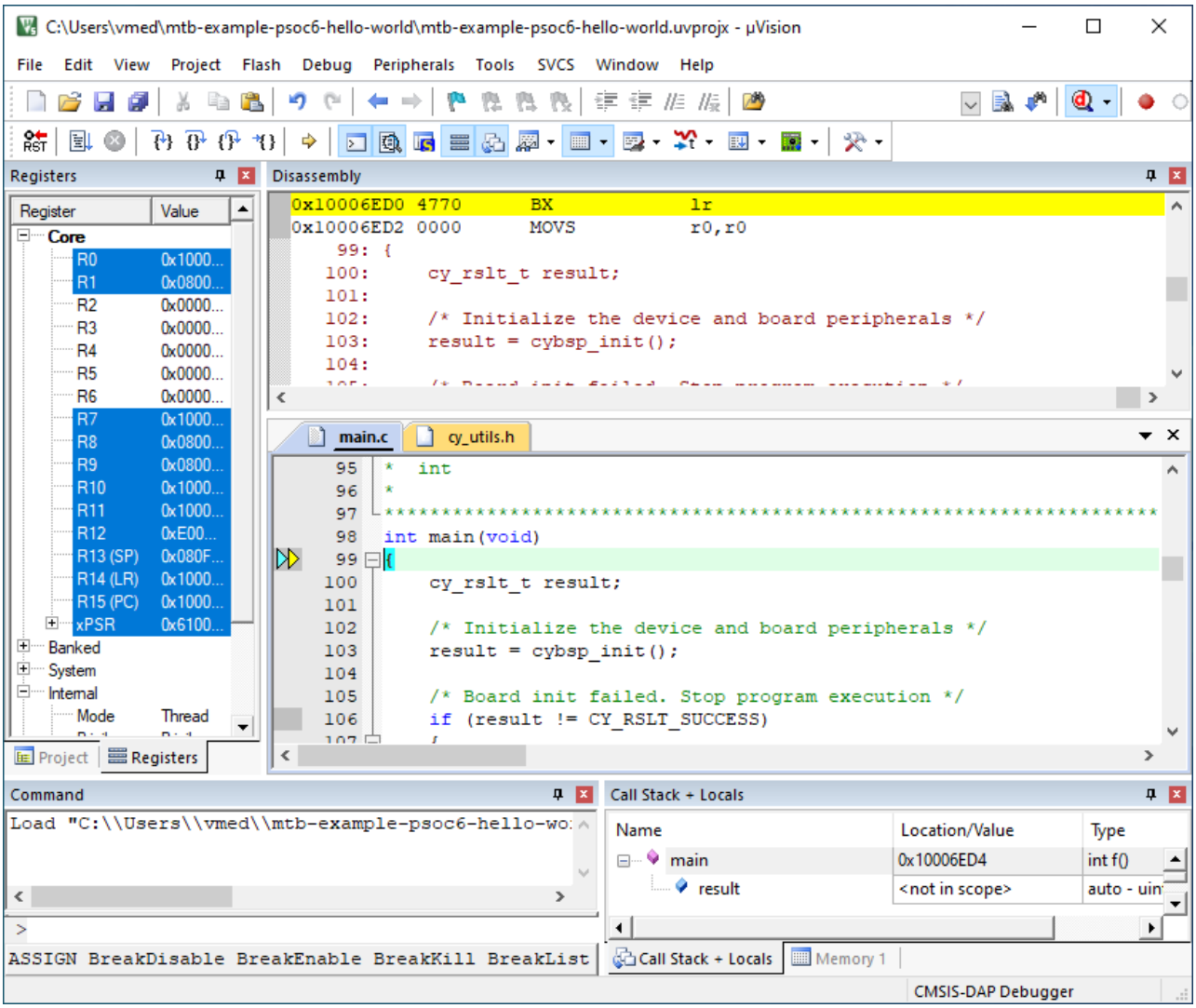
1. Connect the PSOC™ 6 kit to the host PC.

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 fw-loader tool is located in this directory by default:

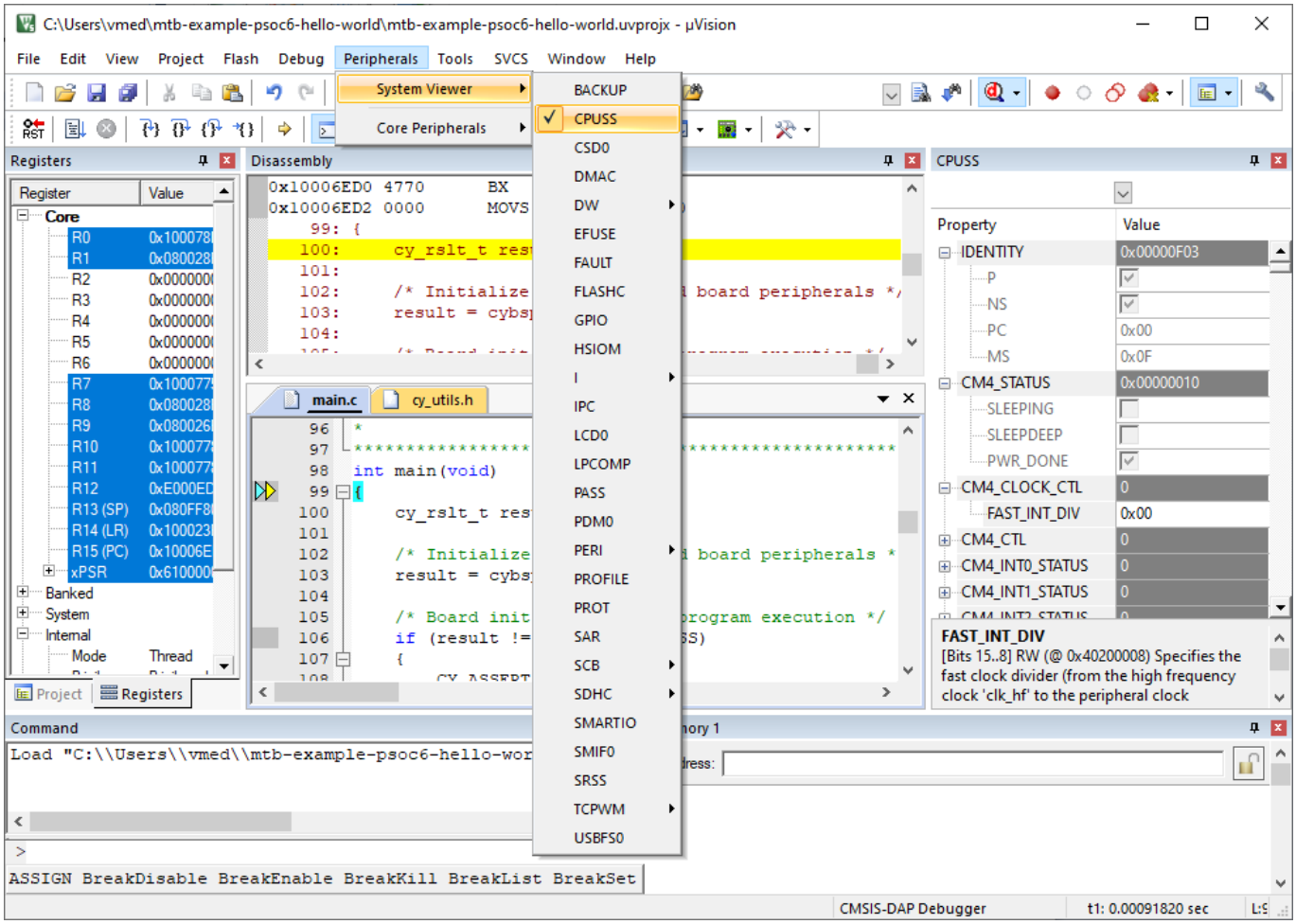
   [install-path]/ModusToolboxProgtools-[version]/fw-loader/bin/

3. Select **Debug > Start/Stop Debug Session**.

6 PSOC™ 4 and PSOC™ 6 single-core application



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

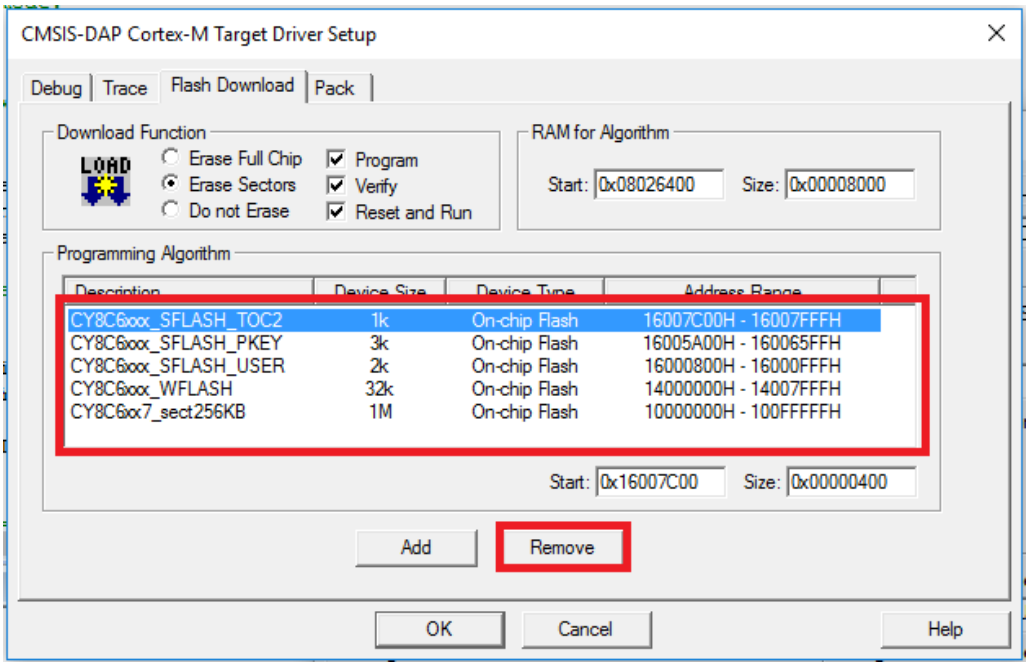


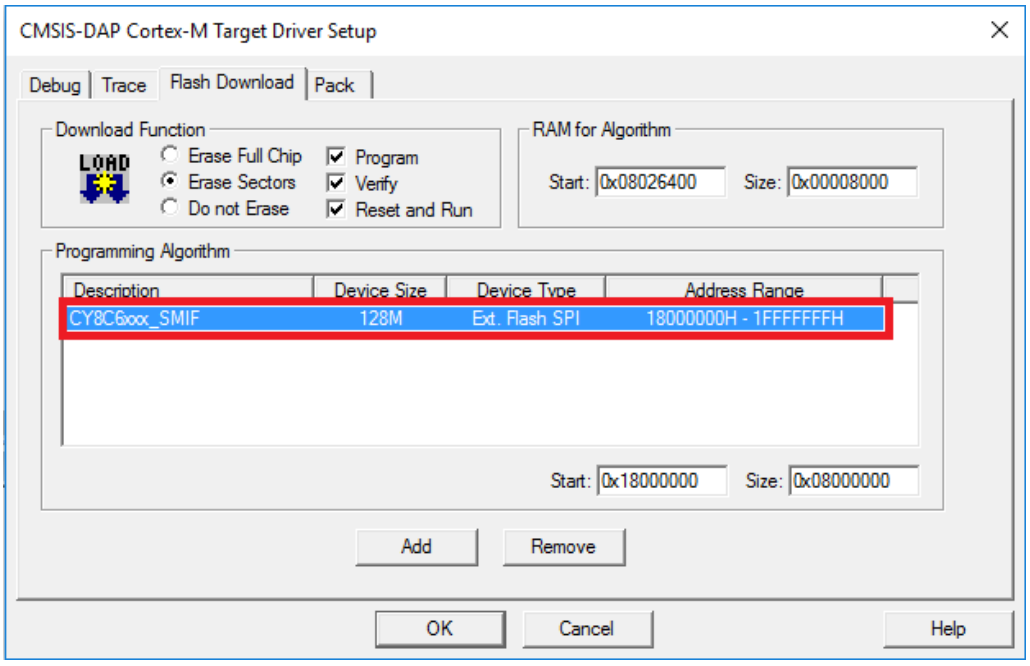
6.2.1 Program external memory

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:





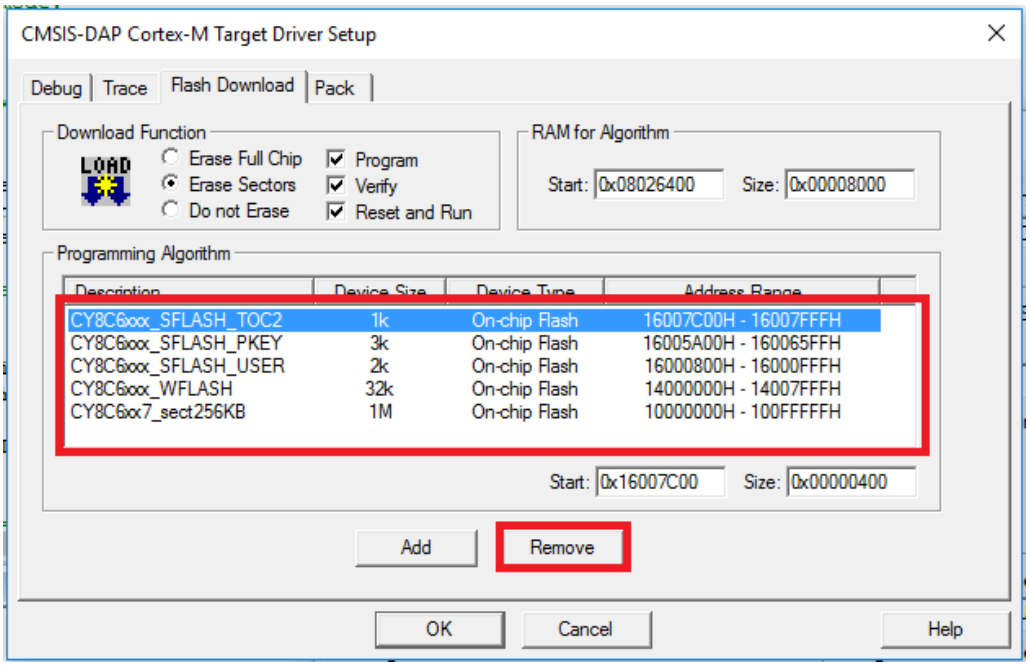
3. Download flash.

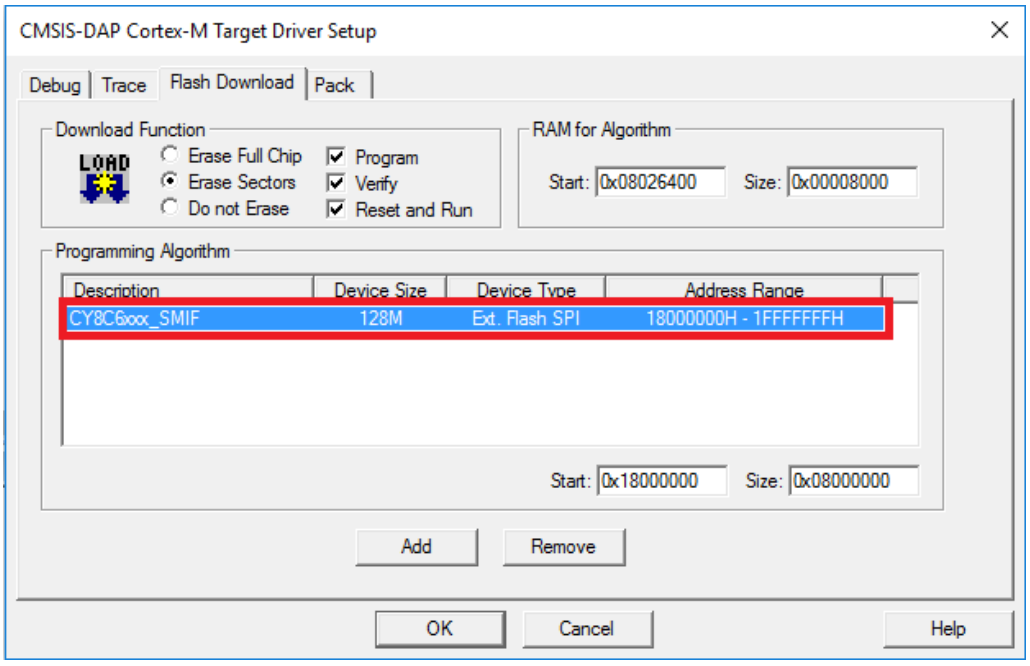
Notice warnings:

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

6.2.2 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:

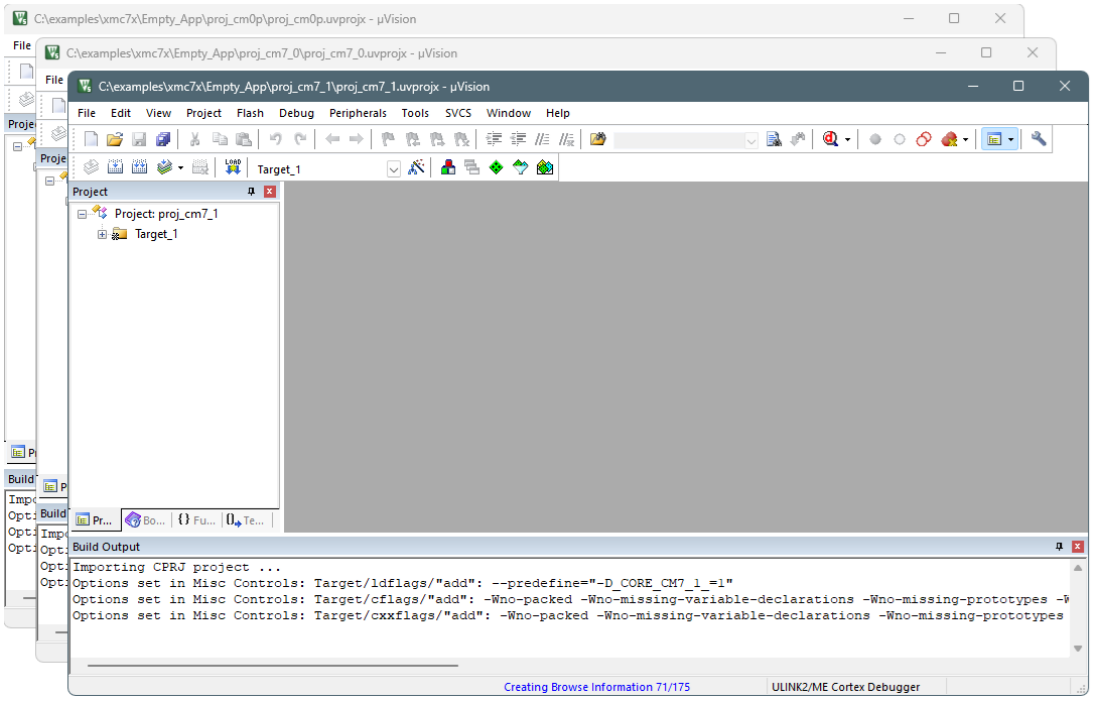




2. Click **Flash > Erase** in menu bar.

7 PSOC™ 6 and XMC7xxx multi-core application

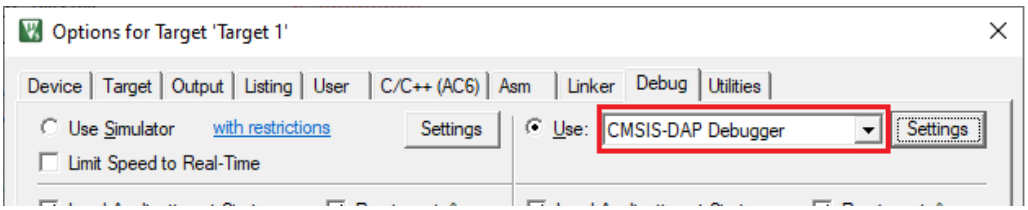
Follow instructions in Create/export application for Keil µVision. After creating all the projects in separate instances, your application should look like this:



Then, use the instructions in this section to configure, build, and debug your application.

7.1 Configure CM0+ project

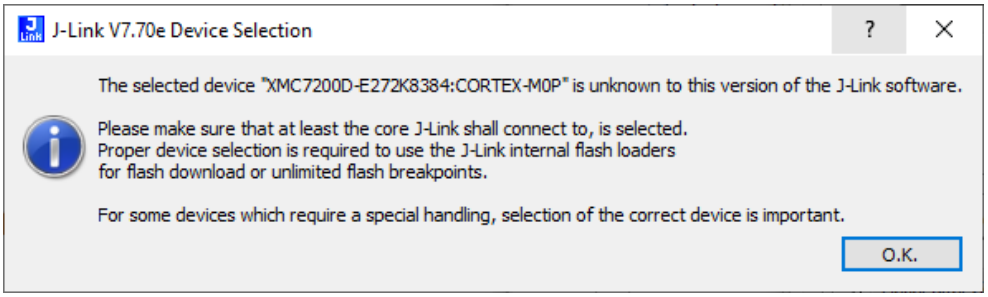
1. Go to **Project > Options for Target <target_name>**, switch to the **Debug** tab, select the applicable debug probe (CMSIS-DAP or J-Link) as shown:



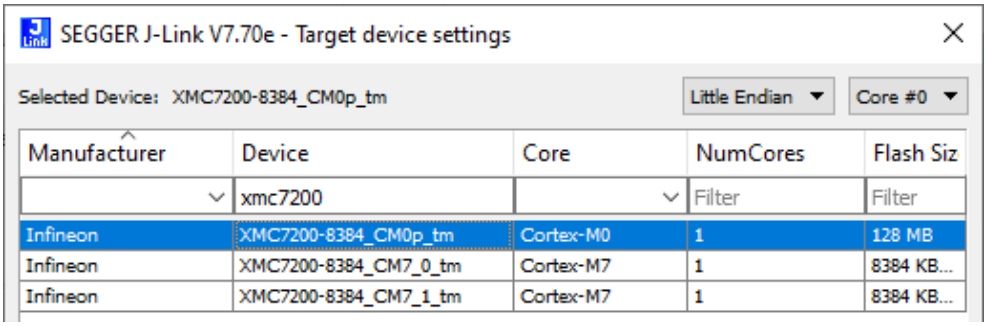
- If using ULINK2, select the **CMSIS-DAP Debugger** option as the debug probe, because the ULINK2 driver does not support multi-core debugging

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

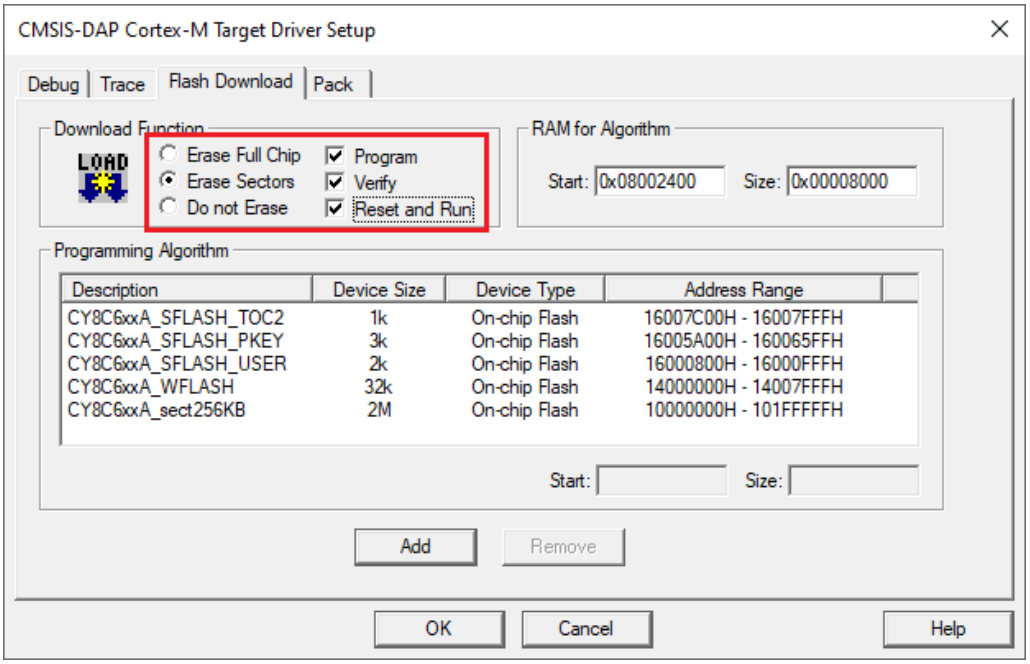
- If you select the J-Link probe, a pop-up window might display reporting that the device is unknown to J-Link software.



- If so, click **OK** 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.



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.

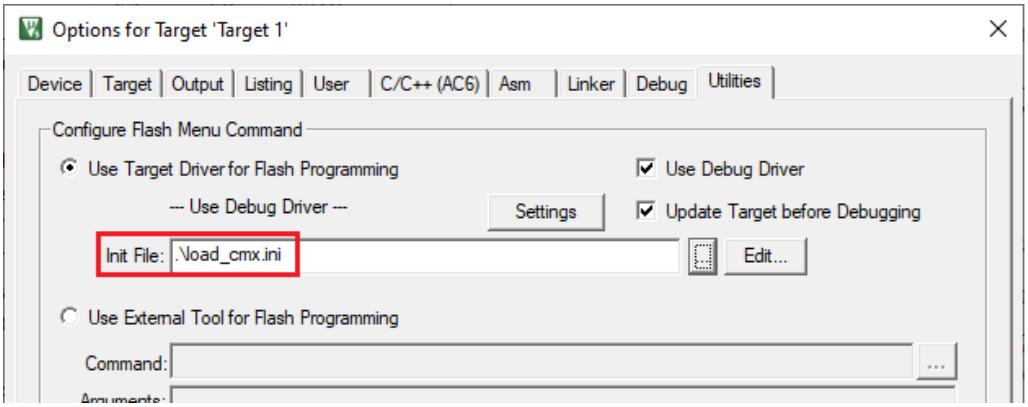
5. Next, configure the project so that it also programs other image(s) from the CM4/CM7 project(s). Do this using the **.ini* file.

- Create a new empty file named load_cmx.ini and save it inside the CM0+ project directory.
- Add a LOAD command with a path to the CM4/CM7 images. For example:

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

- 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 *Init File* edit field.



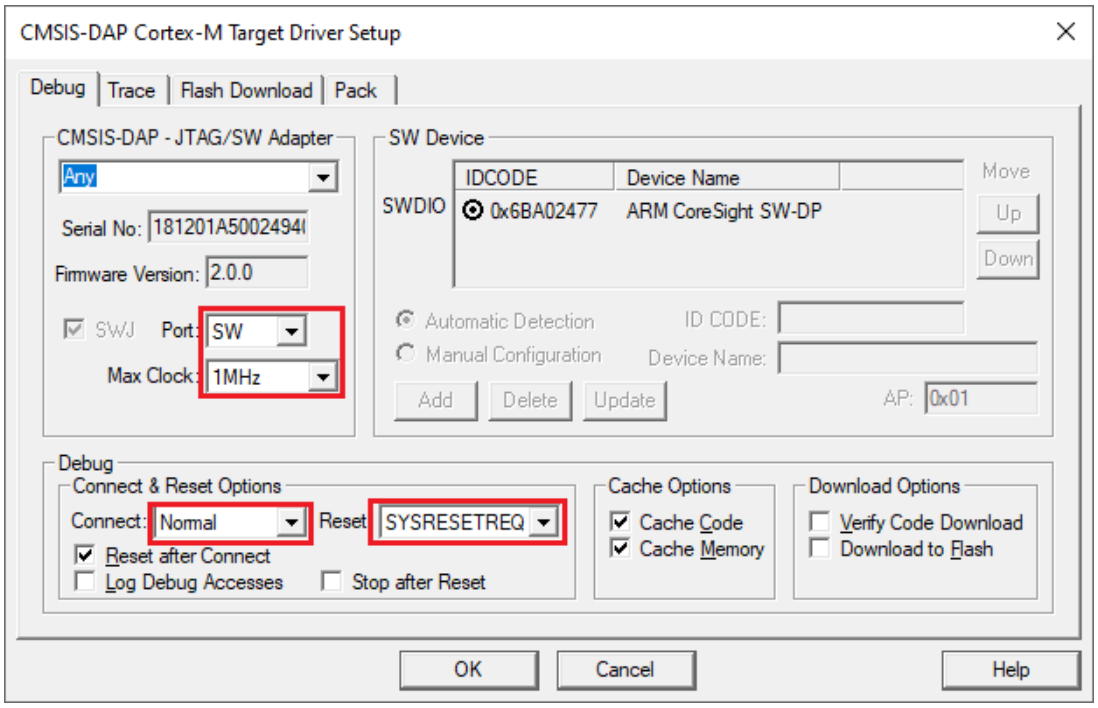
7. Switch to the **Debug** tab, and click the **Settings** button.

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:

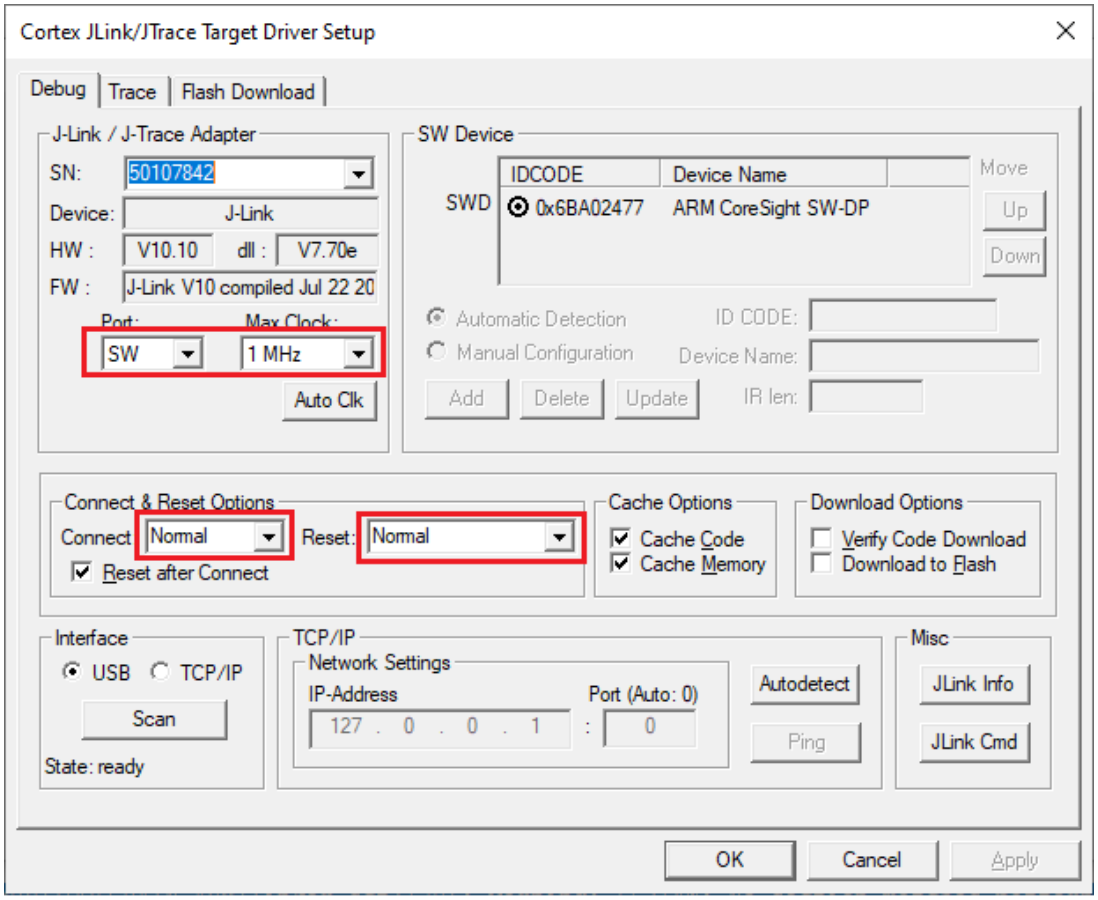
- **Port**: SW
- **Max Clock**: 1 MHz
- **Connect**: Normal
- **Reset**: SYSRESETREQ



J-Link Target Driver Setup

Use the following options:

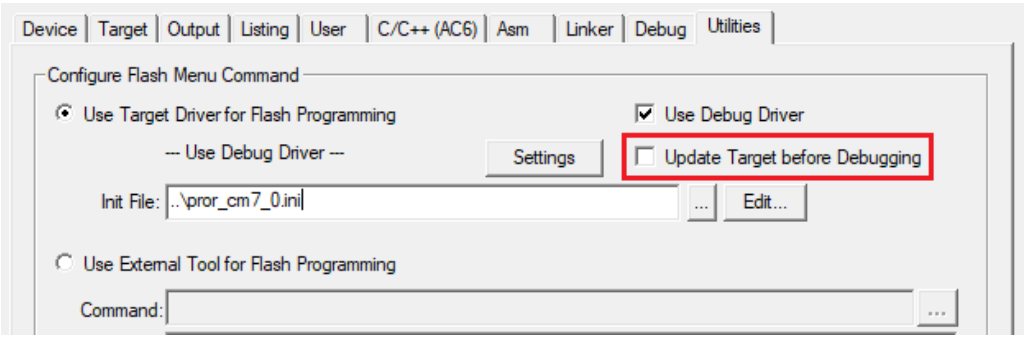
- **Port**: SW
- **Max clock**: 1 MHz
- **Connect**: Normal
- **Reset**: Normal



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

7.2 Configure CM4/CM7 project

1. Open the Options dialog to the **Utilities** tab, and deselect the **Update Target before the Debugging** check box.

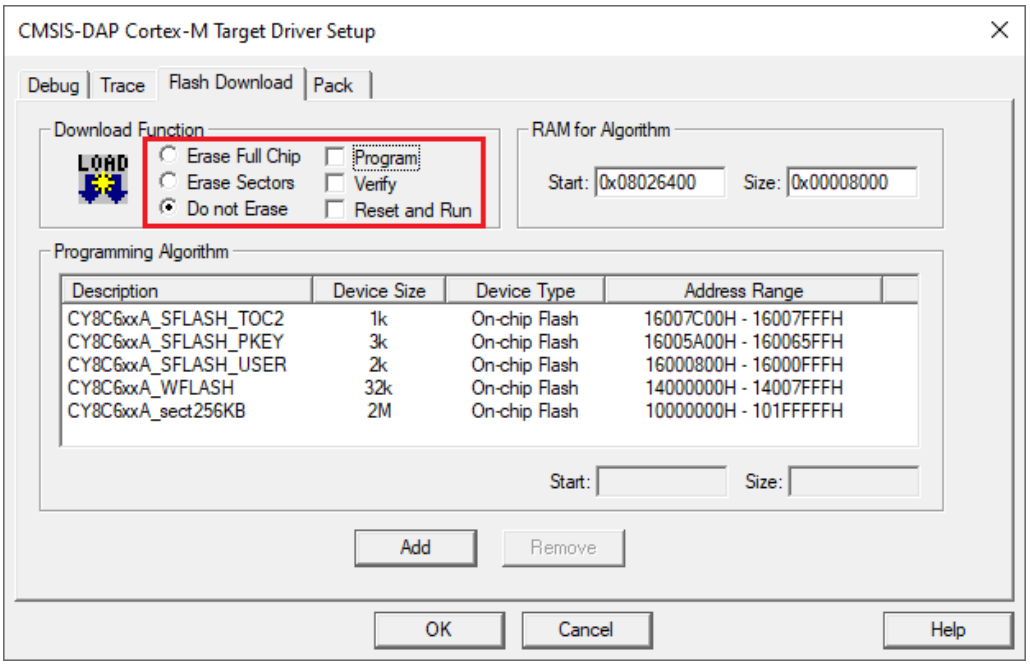


2. Switch to the **Debug** tab, select the applicable debug probe (CMSIS-DAP or J-Link).

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

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

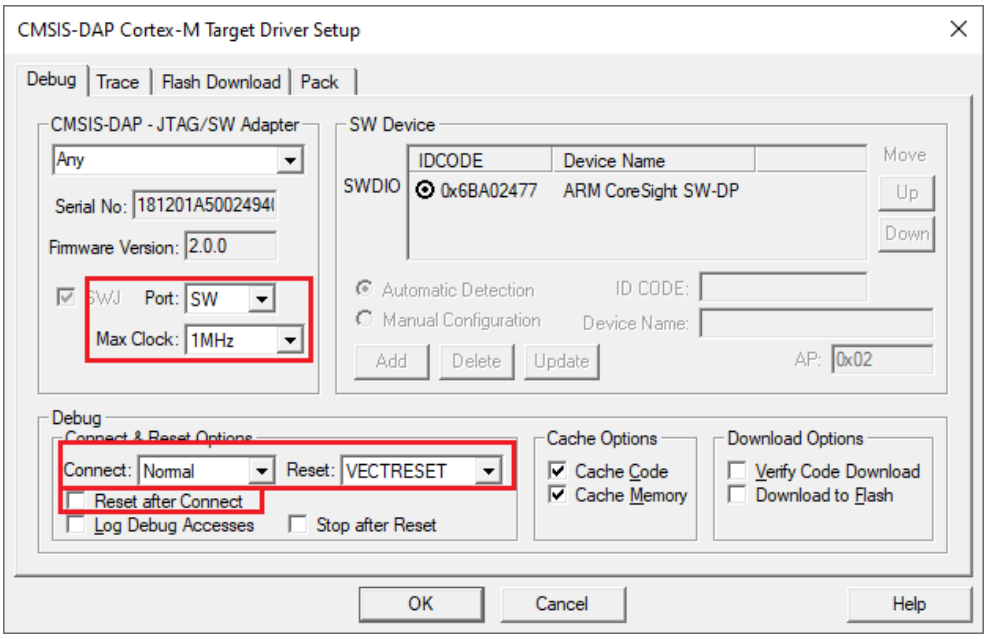
4. On the Target Driver Setup dialog, switch to the **Flash Download** tab, select the **Do not Erase** radio button, and deselect the **Program**, **Verify**, and **Reset and Run** check boxes.



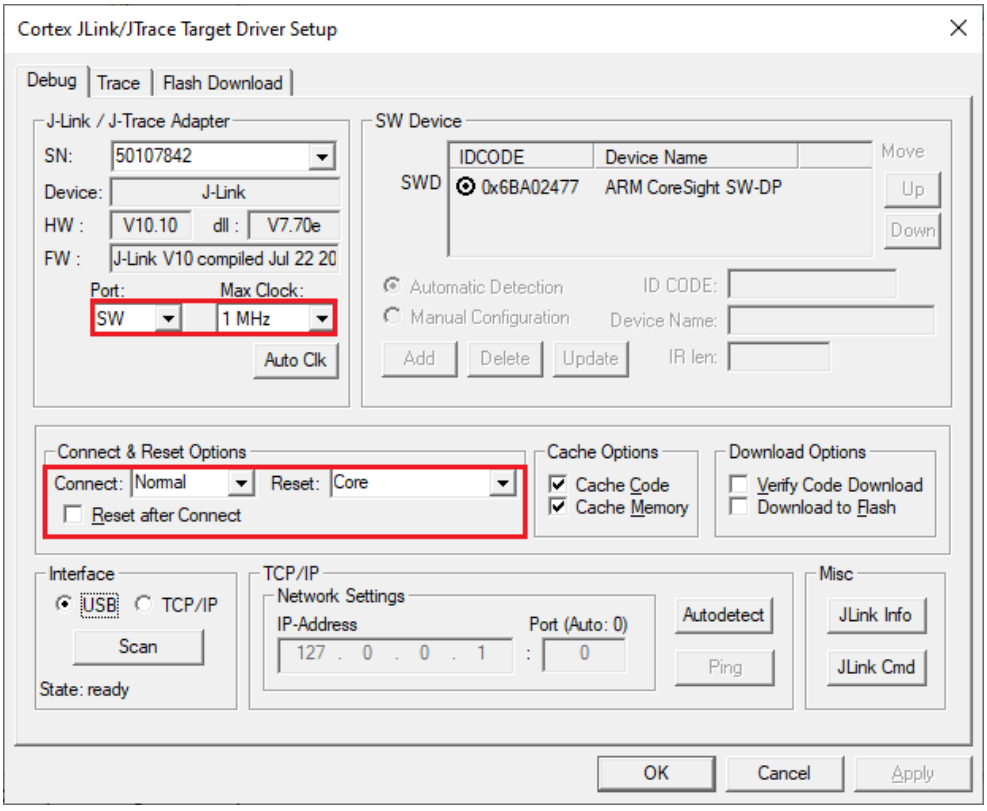
5. Switch to the **Debug** tab.

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



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

7. Save the project(s).

7.3 Building µVision multi-core projects

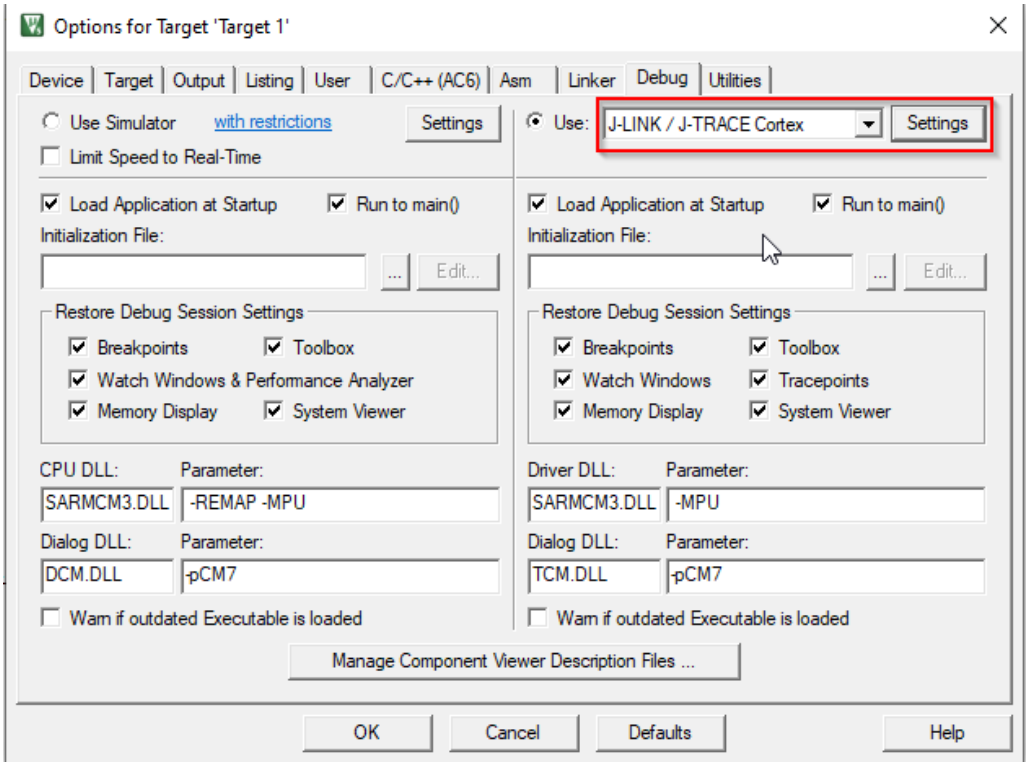
Once all projects are open, go ahead and build each one using **Project > Build target**.

7.4 To use J-Link debugger with XMC7000 devices

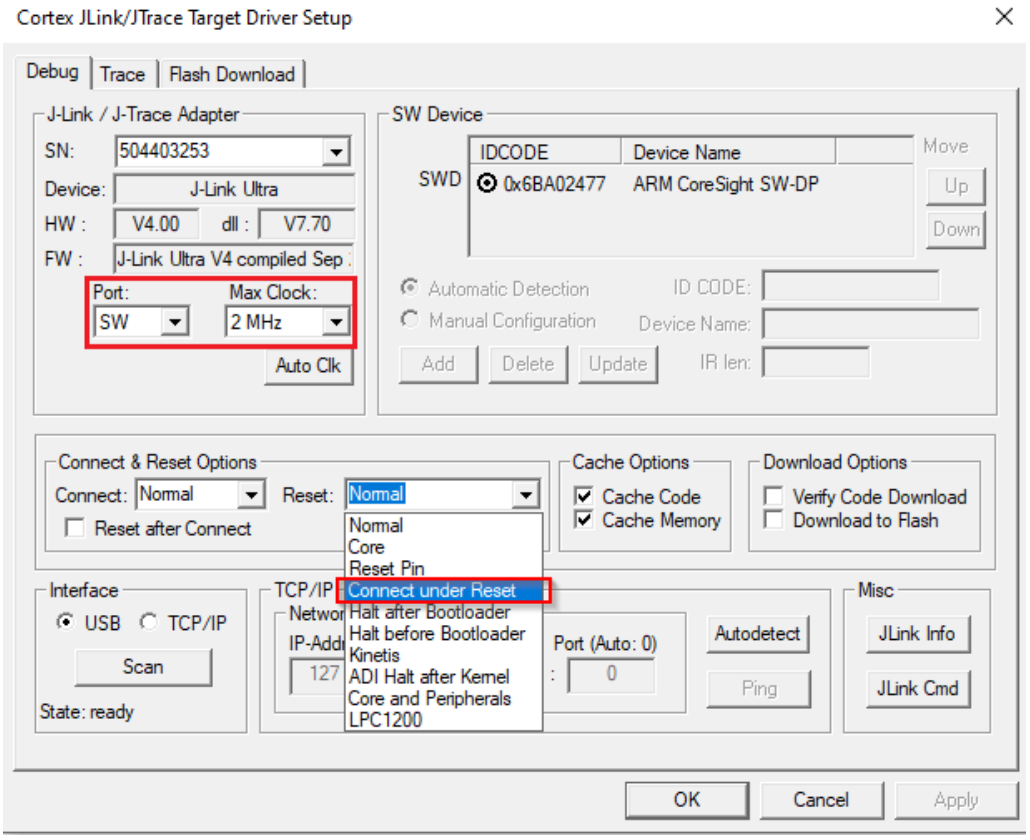
- Program set-up instructions
- Debug set-up instructions

7.4.1 Program set-up instructions

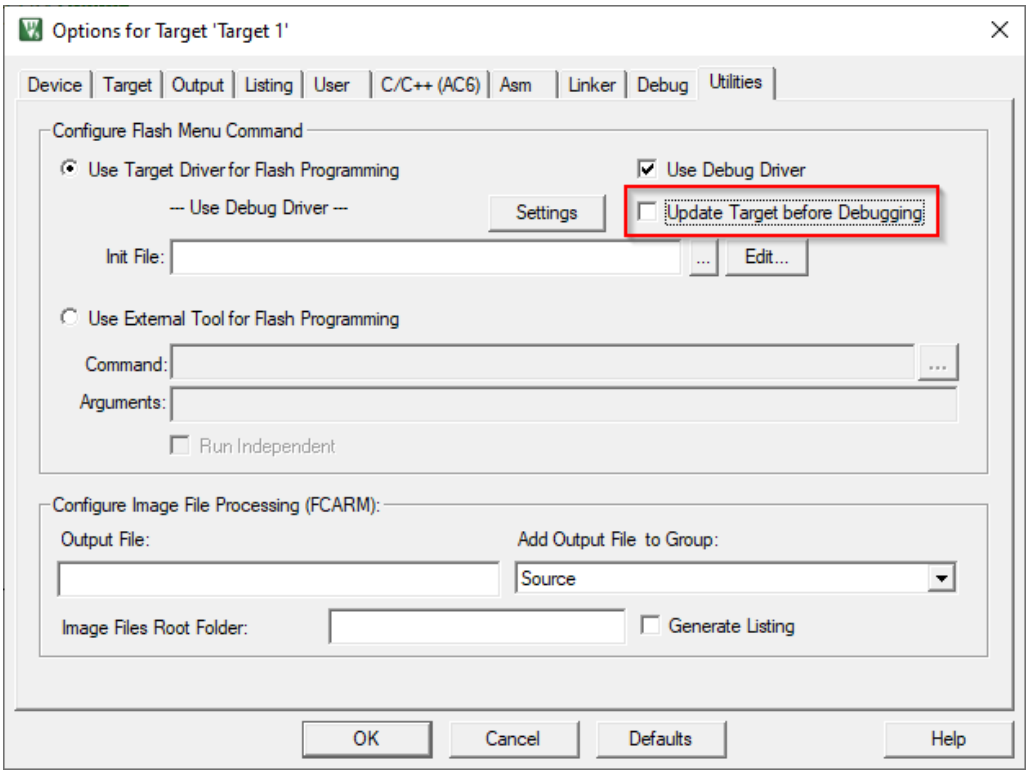
1. Select the **Debug** tab in the Options for Target dialog, select "J-LINK / J-TRACE Cortex" as the debug adapter, and click **Settings**.



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

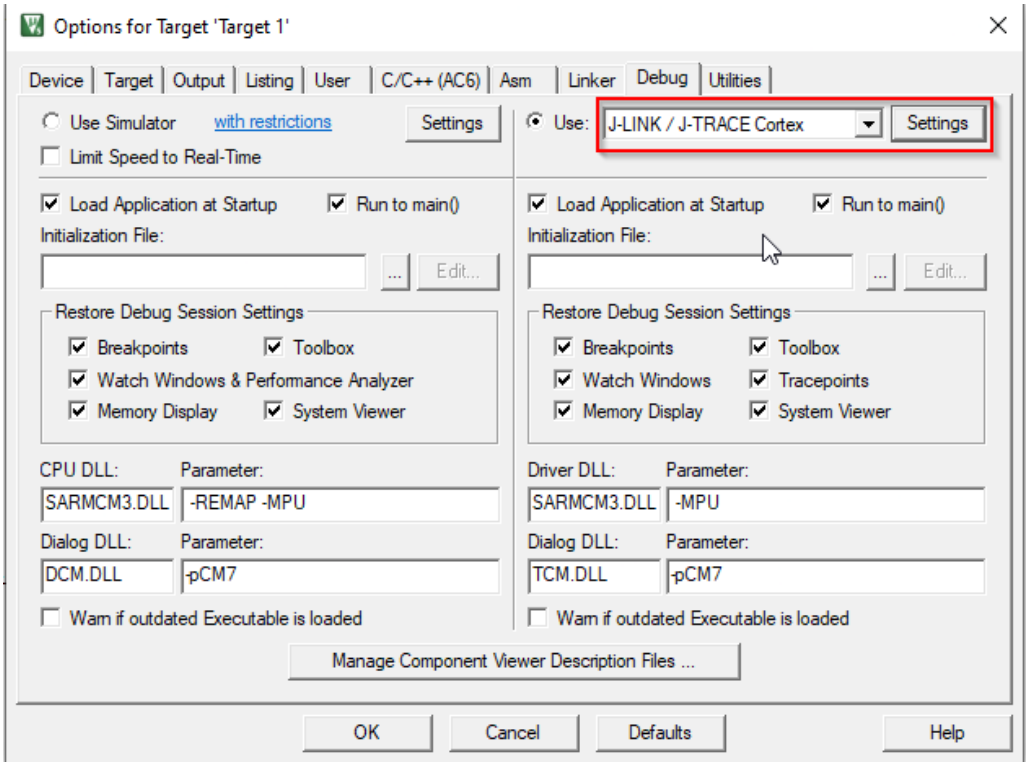


3. Select the **Utilities** tab in the Options for Target dialog and deselect the **Update target before Debugging** check box.



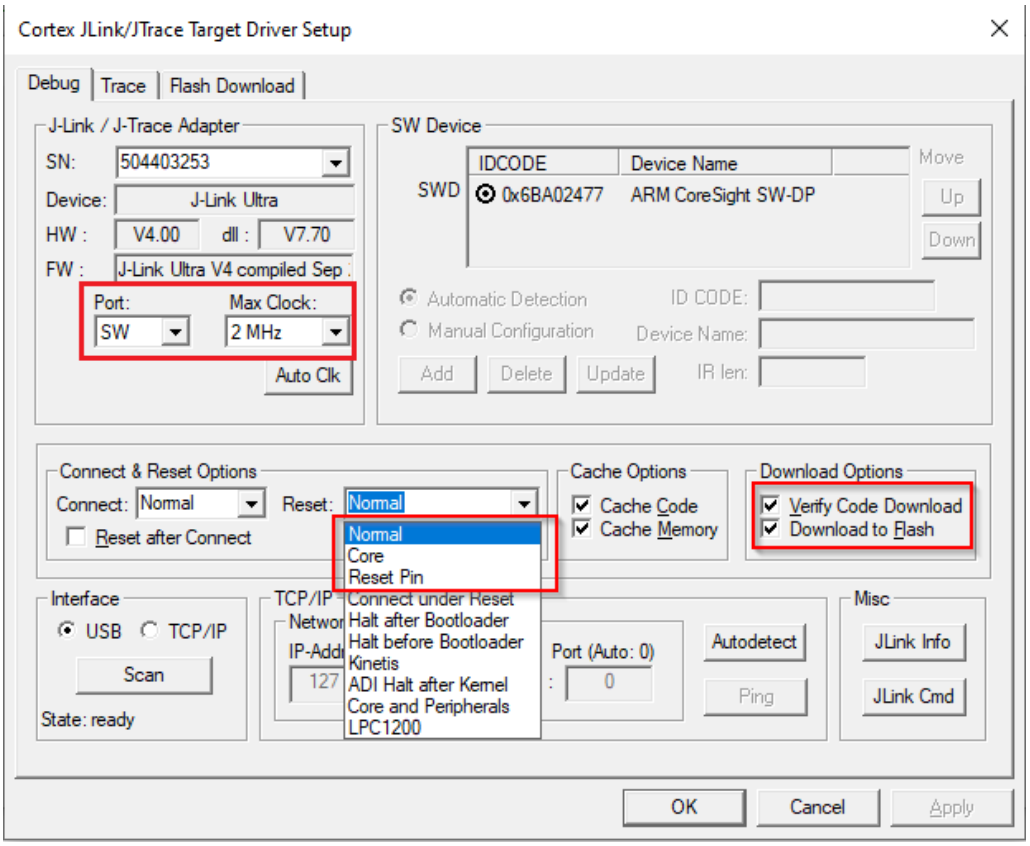
7.4.2 Debug set-up instructions

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:

- **Port**: select "SW"
- **Max Clock**: select 2 MHz
- **Reset** pull-down: select "Normal," "Core" or "Reset Pin"
- **Download Options**: enable "Verify Code Download" and "Download to Flash"



7.5 Launch multi-core debug session

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

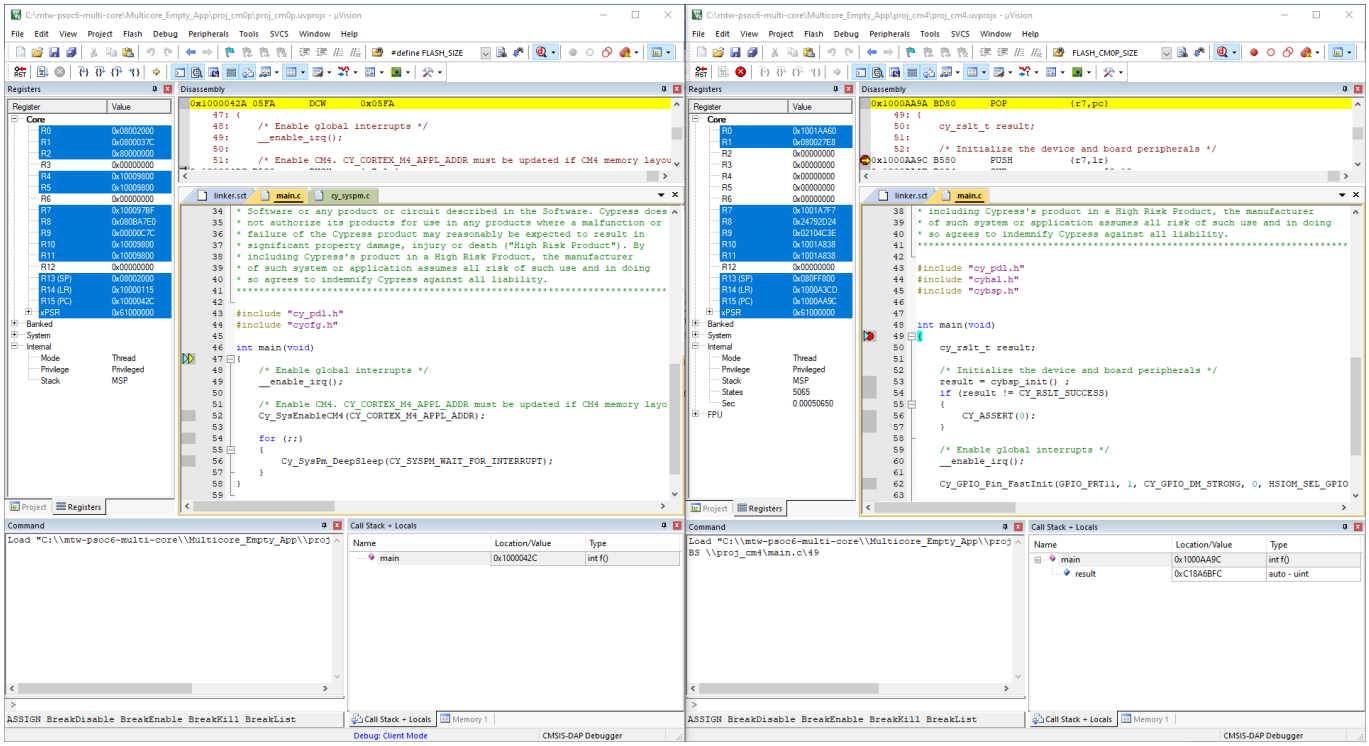
1. Open a µVision IDE session with the project for the CM0+ core and start debugging by pressing **Debug > Start/Stop Debug Session**. This will program all images, reset the target, and halt at the beginning of the CM0+ project `main()`.

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.*

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

7 PSOC™ 6 and XMC7xxx multi-core application



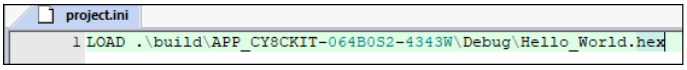
The left side of the screen shows a µVision IDE instance attached to the CM0+ core. The right side shows the CM4 core has not started yet. Once the `Cy_SysEnableCM4()` function on the CM0+ core has been executed, the CM4 will start executing its application. You can step through the code by switching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

8 PSOC™ 64 secure single-core application

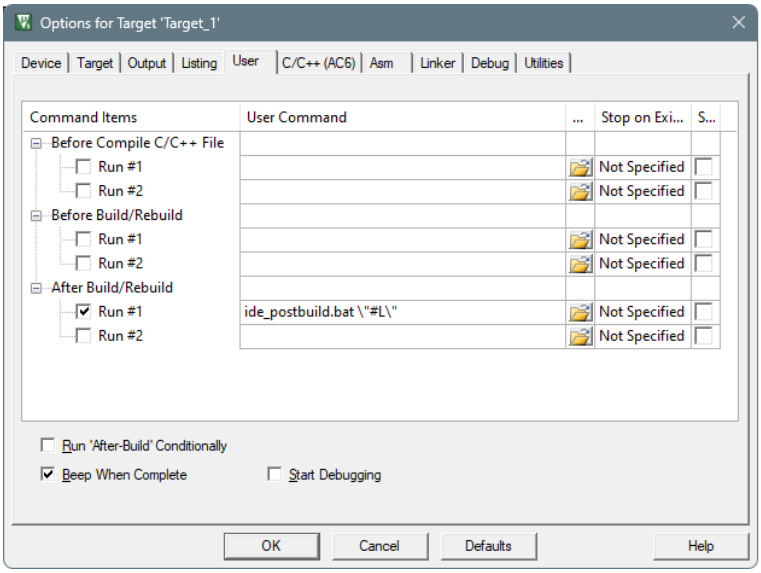
Follow instructions in Create/export application for Keil µVision. The project creation process also generates an *ide_postbuild.bat* file used to execute postbuild commands for the application. When complete, use these instructions to configure, build, and debug your application.

***Note**: For a PSOC™ 64 secure MCU, you must also follow the instructions in the Secure Boot SDK user guide to generate keys and provision the device.*

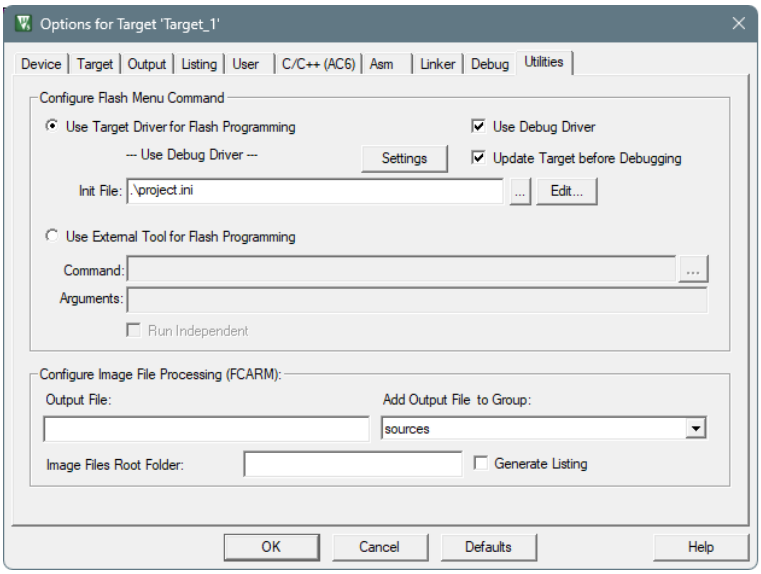
1. Using the Keil µVision IDE, create an empty *project.ini* file in the project root folder.
2. Type `LOAD .\build\APP_[kit-name]\Debug\[project-name].hex` in the *project.ini* file and save the file. Replace [kit-name] and [project-name] with the correct values.



3. Then, select **Project > Options for Target 'Target 1'...** to open the Options dialog.
4. Select the **User** tab and enable the **Run #1** check box under **After Build/Rebuild**. Then, enter `ide_postbuild.bat \"#L\"`.



5. Select the **Utilities** tab and select the *project.ini* file using the [**...**] button.



6. Then, click **OK** to close the options dialog.
7. 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.

9 AIROC™ CYW20829 single-core application

Follow instructions in Create/export application for Keil µVision. Then, use these instructions to configure, build, and debug your application.

1. Create a *program.ini* file in the project root folder and add the following text:

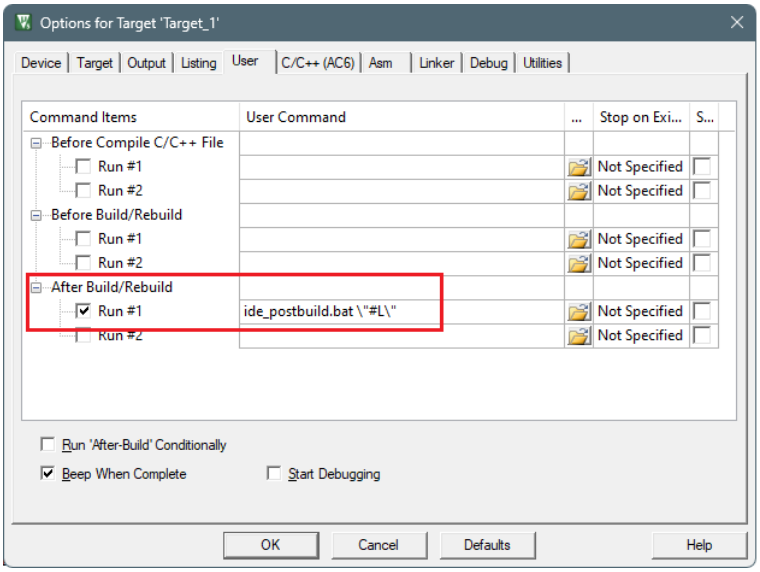
```
LOAD .\build\last_config\mtb-example-empty-app.final.hex
```

2. Create a *debug.ini* file in the project root folder and add the following text:

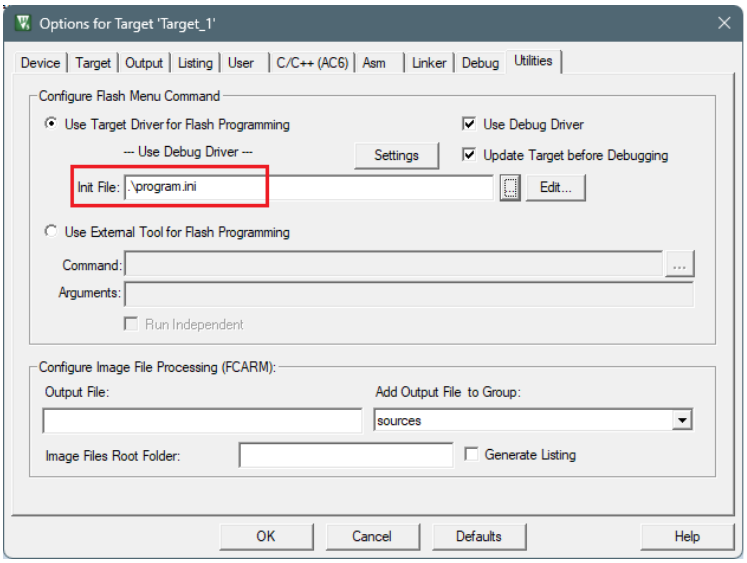
```
LOAD $L%L NOCODE CLEAR INCREMENTAL
g, main
```

3. Open the Options dialog and switch to the **User** tab.
4. Enable the **Run #1** check box located under **After Build/Rebuild**. Then, paste the following command as single line (edit paths as needed).

```
ide_postbuild.bat \"#L\"
```

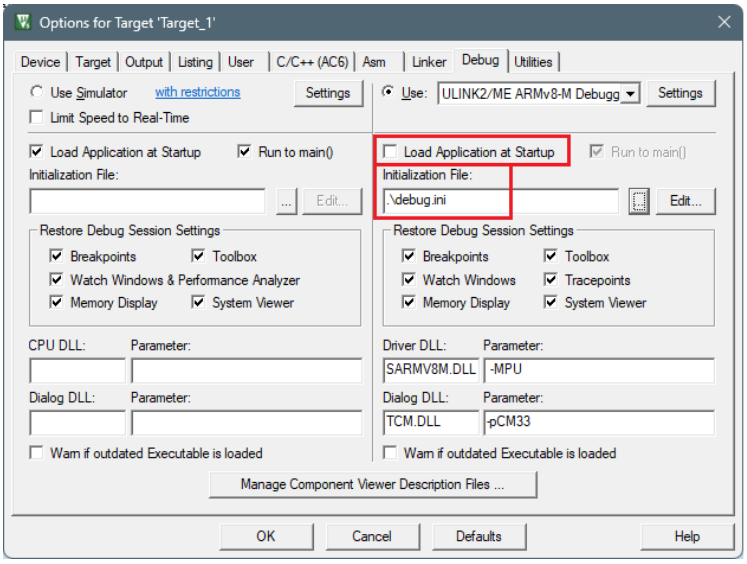


5. Switch to the **Utilities** tab and select the *program.ini* file in the **Init File** field.



6. Switch to the **Debug** tab.

- Make sure that the **Load Application at Startup** check box is not checked.
- Select the *debug.ini* file in the **Initialization File** field.



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

8. On the main menu, select **Project > Build Target** to build the application and execute post-build commands.

After following these steps, you should be able to run **Debug**, **Erase**, and **Program** for the AIROC™ CYW20829 device.

10 Make application changes and re-export

After you have successfully built the application and programmed the device, you may wish to make changes, such as configuring hardware settings for the BSP and the device, as well as adding and updating libraries. To do that, you will need to open a specific ModusToolbox™ tool or configurator, make appropriate hardware or library changes, and then re-export to update the application.

10.1 Configure hardware

Configuring hardware involves using the BSP Assistant tool to change the device and/or configurations. We recommend that you back up your project files first. Then, open the Library Manager using one of the methods described in Opening ModusToolbox™ tools and configurators. Refer to the BSP Assistant user guide for all the details about how to use this tool.

If you just want to modify parts of the existing device such as peripherals, ports/pins, clocks, etc., you can open the Device Configurator from the BSP Assistant, or as described in Opening ModusToolbox™ tools and configurators. For more details about that tool, refer to the Device Configurator user guide.

10.2 Add/update libraries

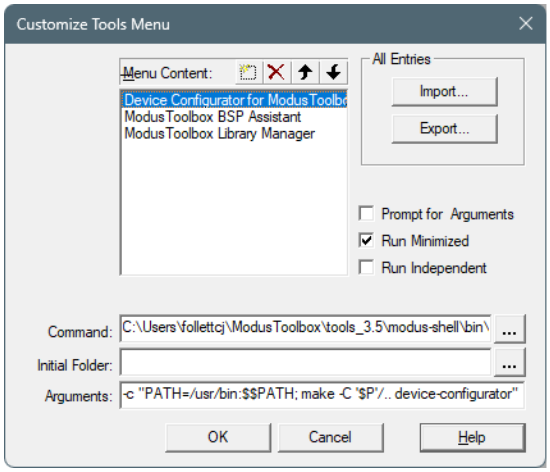
Another way to make changes to your application is by adding and updating various libraries. You do this with the Library Manager. We recommend that you back up your project files first. Then, open the Library Manager using one of the methods described in Opening ModusToolbox™ tools and configurators. For more details about the Library Manager, refer to the user guide.

10.3 Opening ModusToolbox™ tools and configurators

When working with a ModusToolbox™ application in Keil µVision, it is not as convenient to open various tools and configurators as it is using Eclipse or Visual Studio Code. Those IDEs have customized plug-ins or extensions that provide a more integrated experience. Still, there are some ways to open various ModusToolbox™ tools for use with Keil µVision. Here are a few of them:

10.3.1 Configuring Keil µVision

One option to open ModusToolbox™ tools is to configure the Keil µVision **Tools** menu using the **Customize Tools Menu** dialog to add shortcuts to the various tools. This is a one-time configuration for each tool you wish to add. Then, the tools will be available for all your ModusToolbox™ applications running in Keil µVision.



On this dialog, click the **Menu Content "New"** button and enter a name for the tool. Then, complete the following:

- **Command**: `[path-to-modustoolbox-tools]\modus-shell\bin\bash.exe`

  This opens the bash shell from the tools package installation location. You can navigate to the executable using the [**...**] button.
- **Arguments**: `-c "PATH=/usr/bin:$$PATH; make -C '$P' [tool-name]"`

  This passes a command to the bash shell to open the specified `[tool-name]` (for example, device-configurator, bsp-assistant, library-manager, etc.).

  The variable `'$P'` is the µVision project directory, and it is quoted because it returns a Windows-style path, which does not work without the quotes in bash. If the application is multi-core, add `/..` to navigate to one directory higher where the *Makefile* is located.
- **Run Minimized**: This check box is optional. If selected, the bash shell is minimized in the Taskbar running the command to open the tool.

For more details about this dialog, refer to the Keil µVision user guide/help.

10.3.2 Using modus-shell

Another way to open tools and configurators and update the application is to use the modus-shell bash terminal. Open this tool from the installation directory or by typing "modus-shell" in the Windows **Search** box.

As described in the ModusToolbox™ tools package user guide "ModusToolbox™ build system'' chapter, you can run numerous make commands in the application directory. These include launching tools and configurators. In the modus-shell terminal, navigate to the application or project directory containing the *Makefile*, and run a `make [tool-name]` command. For example:

```
cd mtw/example/Hello_World

make library-manager
```

For consistency, the `[tool-name]` is always the name of the tool file name on disk, such as library-manager, bsp-assistant, device-configurator, etc. You can view tool/configurator names by using the `make help` command.

When you open tools and configurators this way, they are opened in context with your application.

10.3.3 Opening directly

Another way to open any ModusToolbox™ tool or configurator is directly by launching it from the installation directory, under *tools_[version]*. On Windows, you can also use the **Search** box and type the name of the tool. After the tool opens, you will have to find the configuration file or project directory from the tool to use it with your application. Each tool and configurator includes a user guide for how to do this.

10.4 Re-export to update application

After running a tool or configurator to make some change, you will need to update the application in Keil µVision using the `make uvision` command with the modus-shell terminal.

***Note**: It's very important to pay attention to the name of your cprj file(s). The name is always the one you specified when creating the project. However, if you exported from another IDE **without** using the `CY_IDE_PRJNAME` variable, the name will come from the project Makefile, which might be different than the cprj file name..*

10 Make application changes and re-export

The basic command is as follows:

```
make uvision CY_IDE_PRJNAME=[existing-cprj-name] TOOLCHAIN=ARM
```

After running the command, verify the *cprj* file(s) is(are) updated, and then double-click to overwrite the *uprojx* file(s). When you view the project in Keil µVision, you will see the updated file structure.

Revision history

Revision	Date	Description
**	2023-05-15	New document.
*A	2023-06-02	Removed obsolete instructions for customizing linker scripts.
*B	2024-01-25	Updates for version 3.2.0. Removed Python. Add instructions for CYW20829 devices. Added link for ETM/ITM trace instructions.
*C	2024-09-27	Updates for version 3.3.0.
*D	2024-12-02	Added instructions to configure a PSOC™ Control C3 device.
*E	2024-12-06	Updates for version 3.4.0.
*F	2025-03-25	Updates for version 3.5.0.
*G	2025-04-28	Separated instructions per each device type.
*H	2025-09-03	Updates for version 3.6.0, and added instructions for a PSOC™ Edge E84 device.
*I	2025-09-24	Minor update to PSOC™ Edge E84 multi-core instructions.
*J	2025-12-12	Updates for version 3.7.0.
*K	2026-01-13	Updated instructions for AIROC™ CYW20829 devices.
*L	2026-03-21	Updates for version 3.8.0; simplified instructions for post-build settings; added link to ITM/ETM Tracing application note for PSOC™ Edge and PSOC™ Control.
*M	2026-04-27	Updated instructions for PSOC™ Edge to simplify the process.

Trademarks

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

Edition 2026-04-27

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-ggj1743010908890

Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. 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) with respect to any and all information given in this application note.

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.