CYPRESS SEMICONDUCTOR CORPORATION
Internal Correspondence

Date: 11/12/2019 **WW:** 1946
To: **WHAO**
Author: CHMA
Author File#: CHMA#44
Subject: **MTB2.0 Dual-Core Support Trial – Create Project via Template**
Category:
Distribution: WHAO; XITO; JRTI; ZQLI; MIQI; CCTA; KOZU; HAXI; LWAX

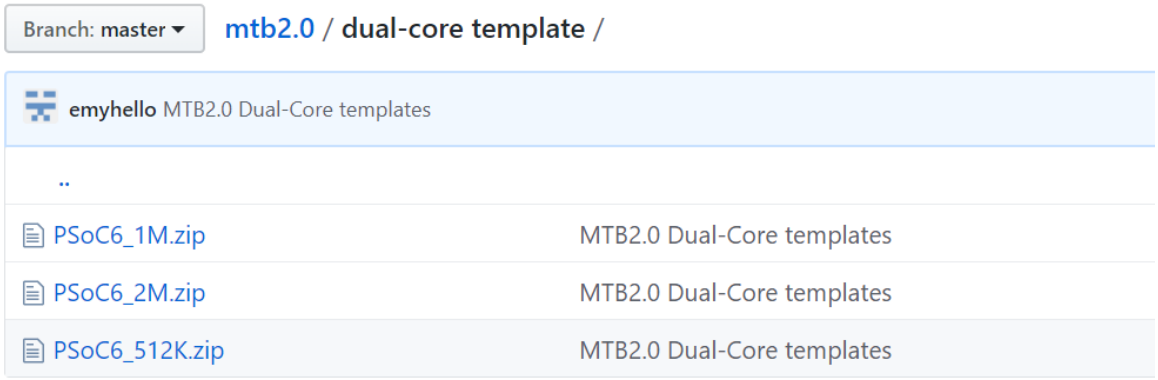
Summary
The memo documents the detailed steps of how to create a dual-core supported application project for specific MPN via the template on MTB2.0.

Feature
- CM0p and CM4 share one Device Configurator tool
- Dual-Core debugging is supported
- Auto-generate the merged final.hex and final.elf files
- Easy to switch the MCU part number
- Fixed EEPROM/SMIF etc. limitation on CM0p core

Prerequisites
ModusToolbox 2.0**:** Build ID#1703
https://www.cypress.com/products/modustoolbox-software-environment

Template:
https://github.com/emyhello/mtb2.0.git

Template Introduction
1) The template consists of 3 zip files, each one corresponding to a PSoC6 family.

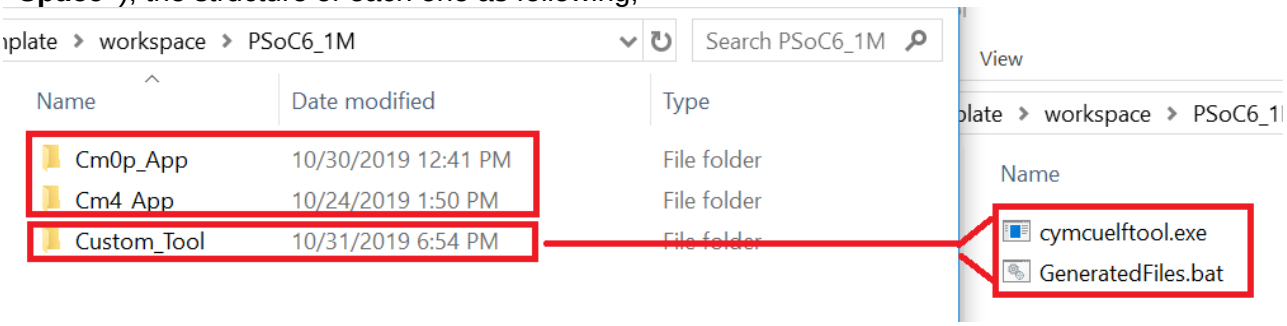


PSoC6_1M.zip: suitable for 1M Flash size PSoC6 MCU, based on CY8CKIT-062-BLE (CY8C6347BZI-BLD53) kit.

PSoC6_2M.zip: suitable for 2M Flash size PSoC6 MCU, based on CY8CPROTO-062-4343W (CY8C624ABZI-D44) kit.

PSoC6_512K.zip: suitable for 512K Flash size PSoC6 MCU, based on CY8C6245LQI-S3D72.

2) Un-compressing the template to your workspace (**the path cannot contain "Space"**), the structure of each one as following,



Cm0p folder: CM0p core's project

Cm4 folder: CM4 core's project

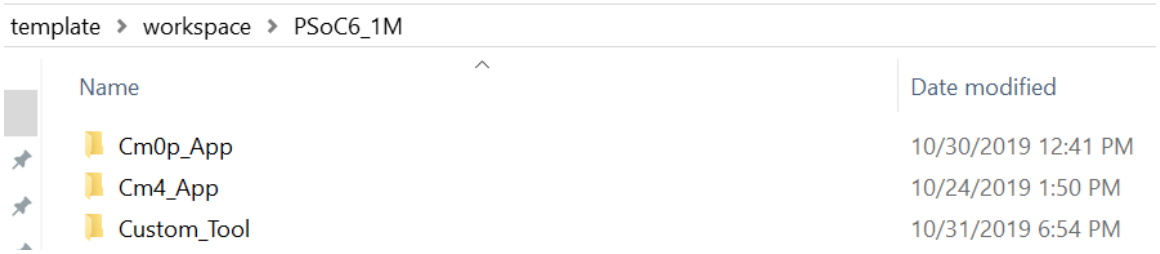
Custom_Tool folder: It consists of 2 files.

- cymcuelftool.exe is an official tool, which is used to merge CM0p and CM4 core images then generate the final image.
- GeneratedFiles.bat is a batch file, it invokes cymcuelftool to generate the final.hex and final.elf files.

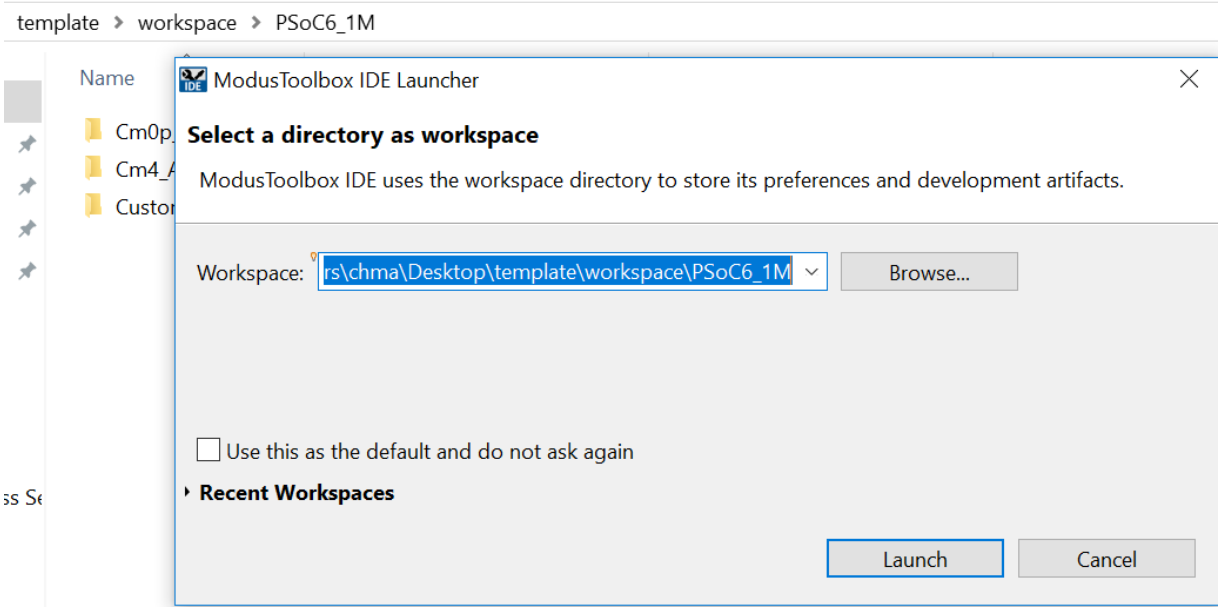
Details

1. Import the template to your workspace

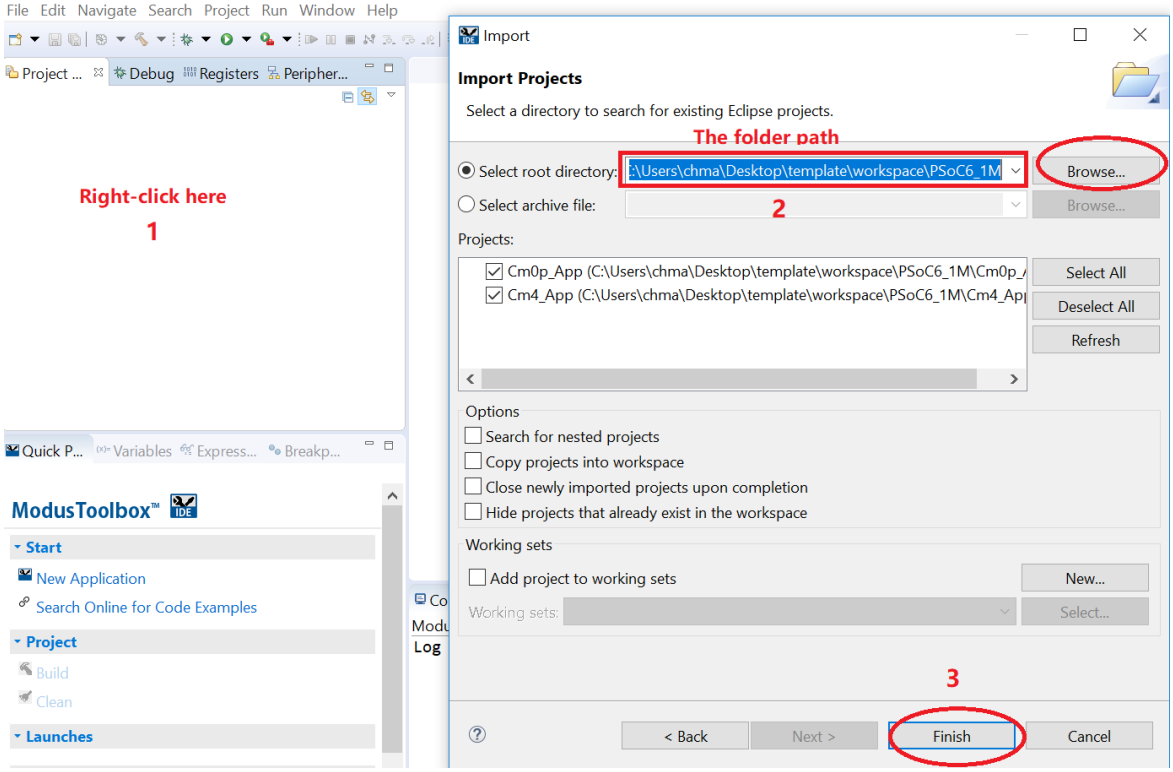
1) Extract the files from PSoC6_1M.zip



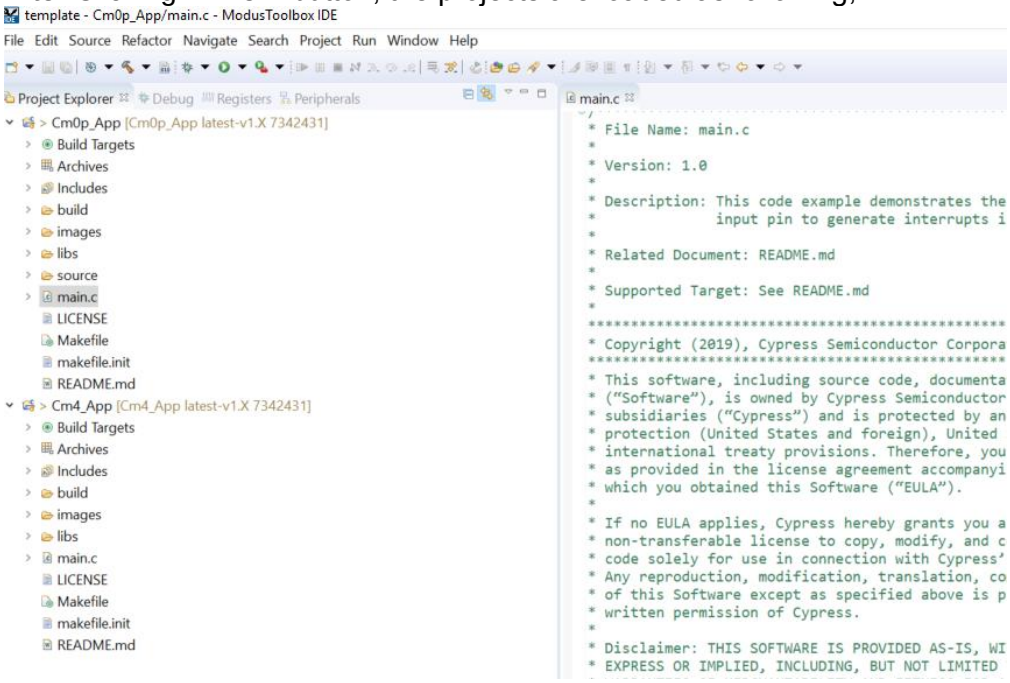
2) Launch MTB2.0 and set the workspace to previous folder



3) Right-Click on the blank space in “Project Explorer”, and click “Import….” Option, then select “General->Existing Projects into Workspace” as following,



4) After Clicking “Finish” button, the projects are loaded as following,

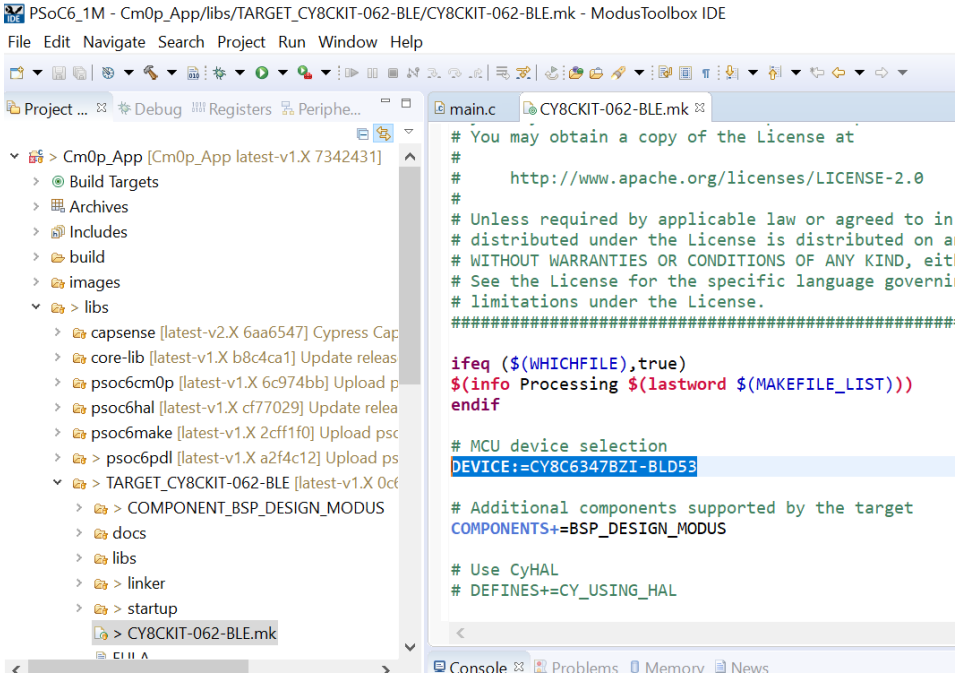


2. Switch the MCU part number

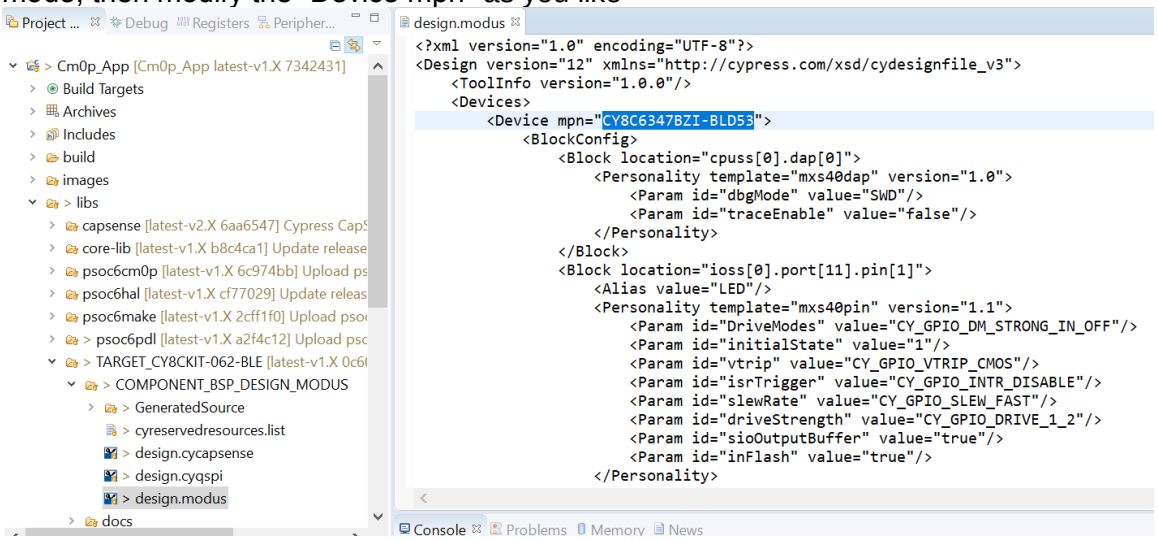
There have 3 places you need to modify.

Note: Before switching MPN, be sure you select the correct template.

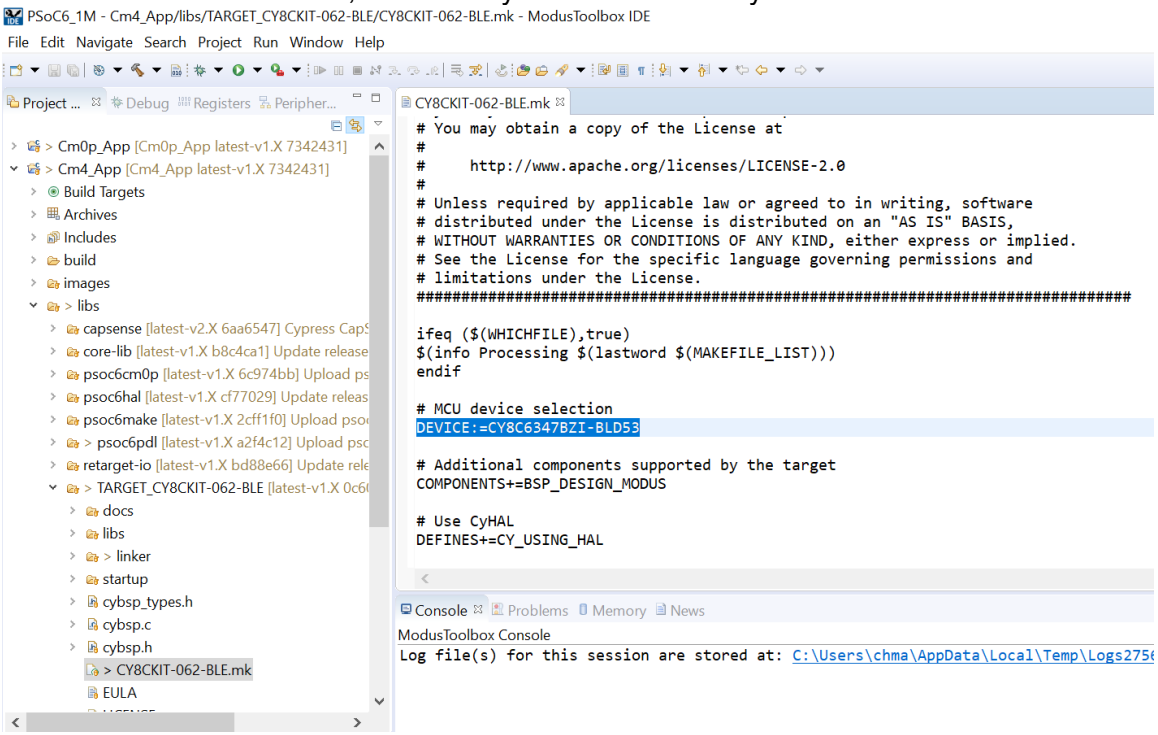
1) Open the \Cm0p_App\libs\TARGET_CY8CKIT-062-BLE \CY8CKIT-062-BLE.mk file with edit mode, then modify the “DEVICE” as your like.



2) Open the \Cm0p_App\libs\TARGET_CY8CKIT-062-BLE\COMPONENT_BSP_DESIGN_MODUS\design.modus file with edit mode, then modify the “Device mpn” as you like

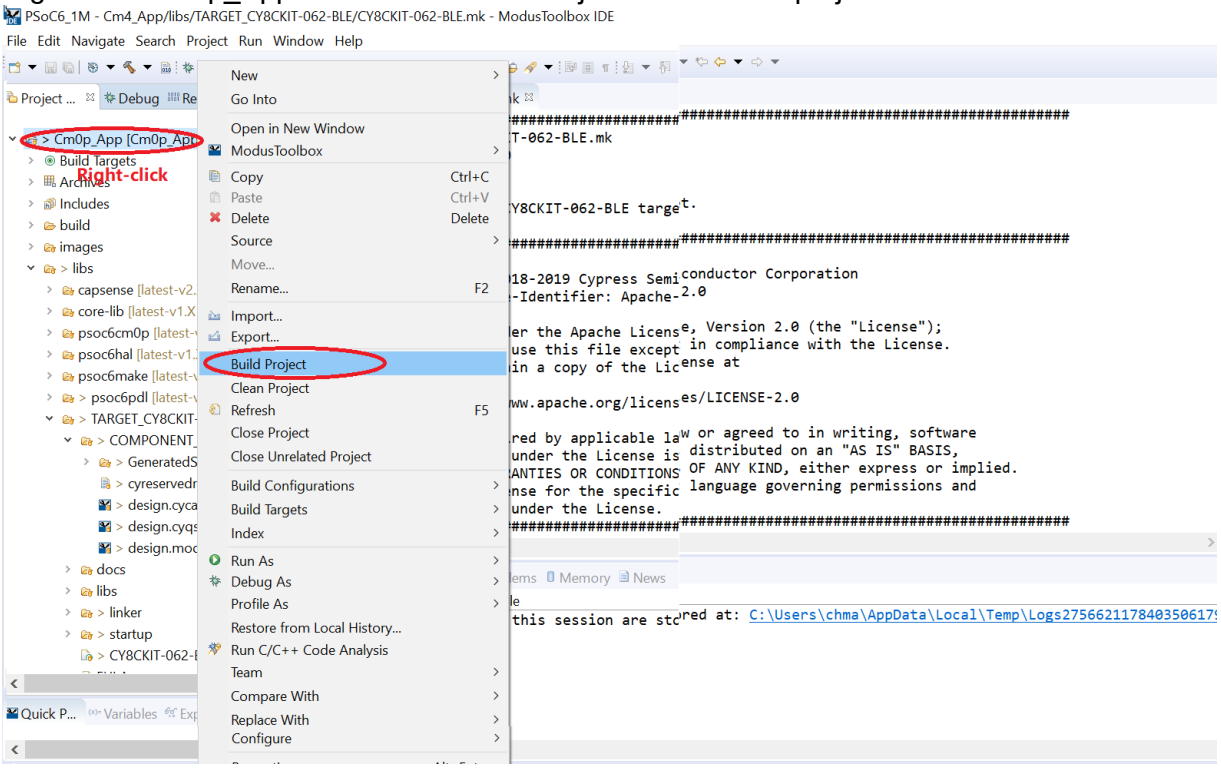


3) Open the \Cm4_App\libs\TARGET_CY8CKIT-062-BLE \CY8CKIT-062-BLE.mk file with edit mode, then modify the “DEVICE” as your like.

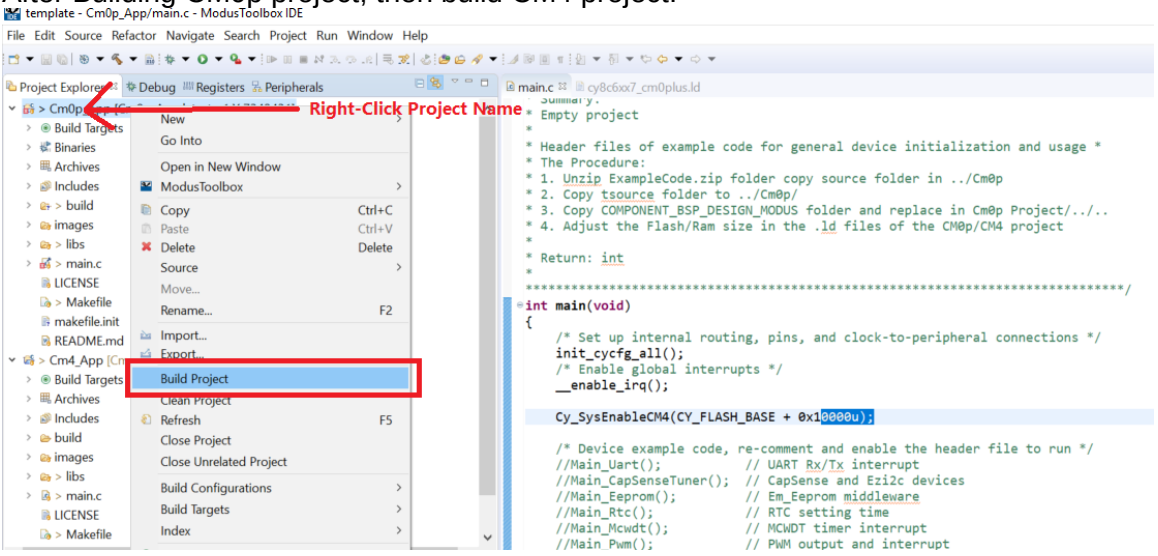


3. Build the projects

1) Right-click Cm0p_App then select “Build Project” to build the project.



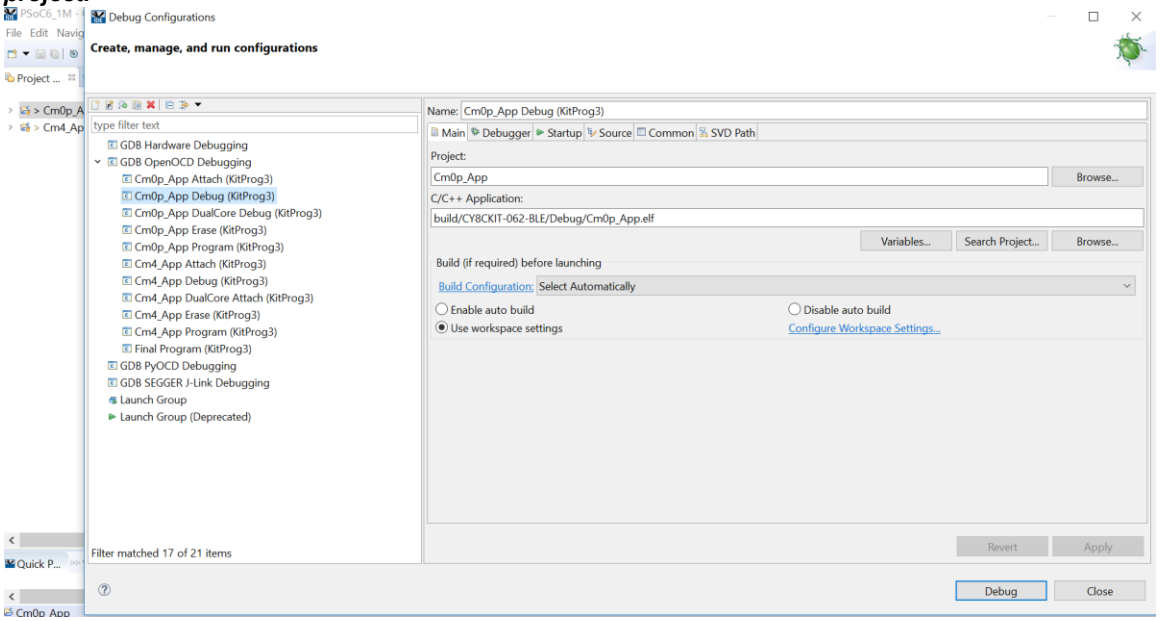
2) After Building CM0p project, then build CM4 project.



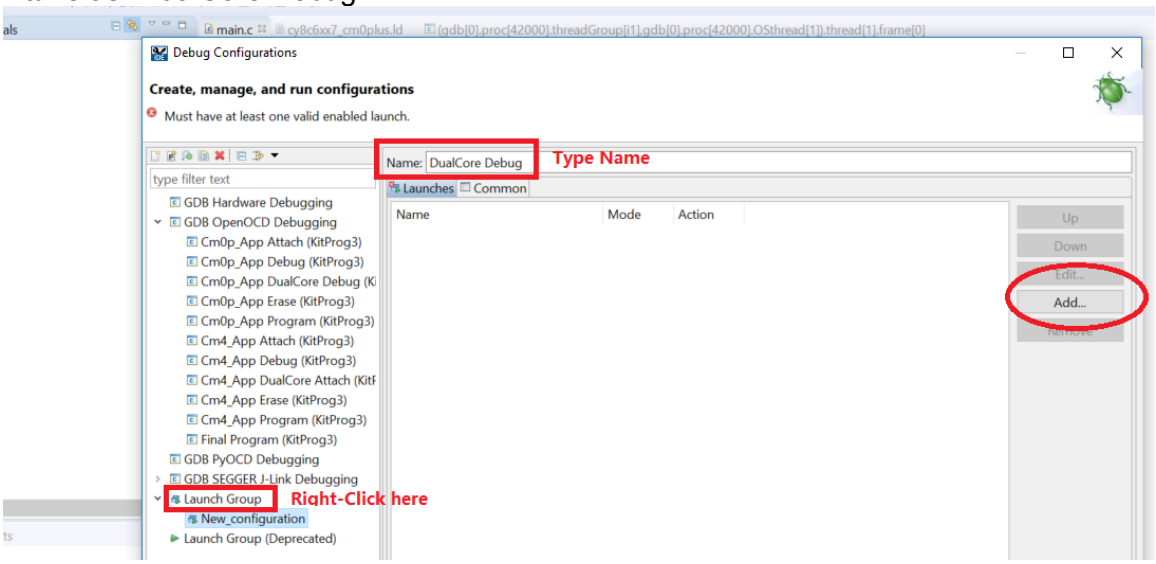
CM0p build output: \Cm0p_App\build\CY8CKIT-062-BLE\Debug
CM4 build output: \Cm4_App\build\CY8CKIT-062-BLE\Debug
Final merged files: \Cm4_App\build\CY8CKIT-062-BLE\Debug

4. Debug Projects

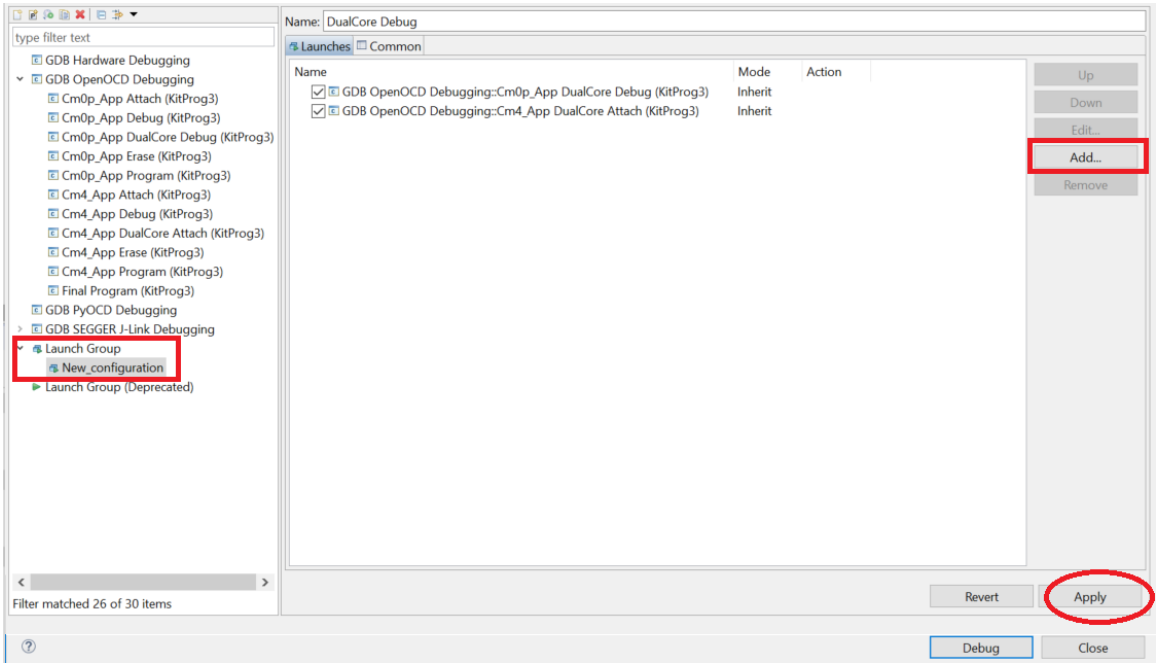
1) Open “Run” option of IDE to open “Debug Configurations…”, you can debugging/programming MCU here.
Note: the “Final Program (KitProg3)” will program whole MCU both CM0p and CM4 project.



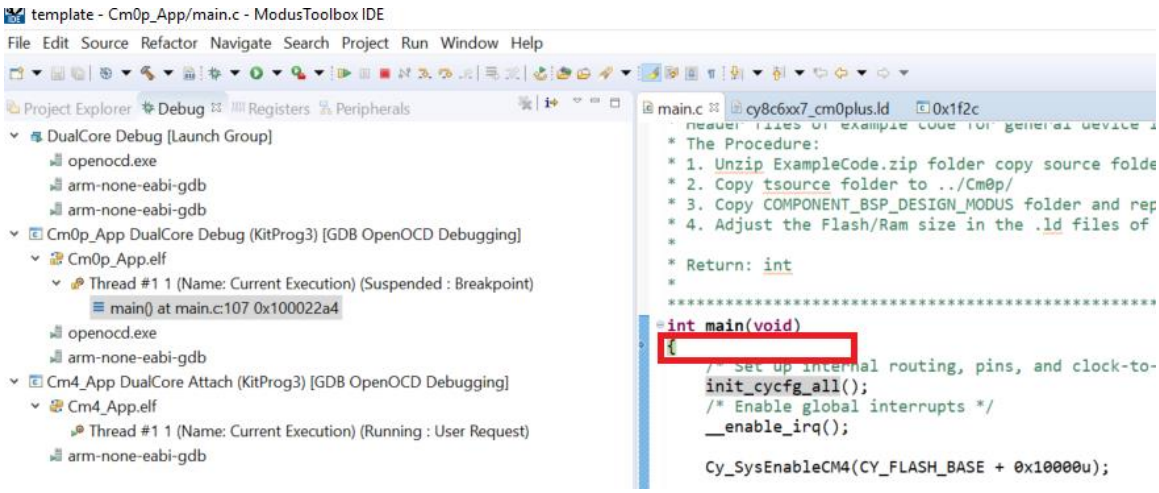
2) Dual-Core debugging
   a. Righ-click “Launch Group”, then select “New_configuration”, type the Name as “DualCore Debug”.



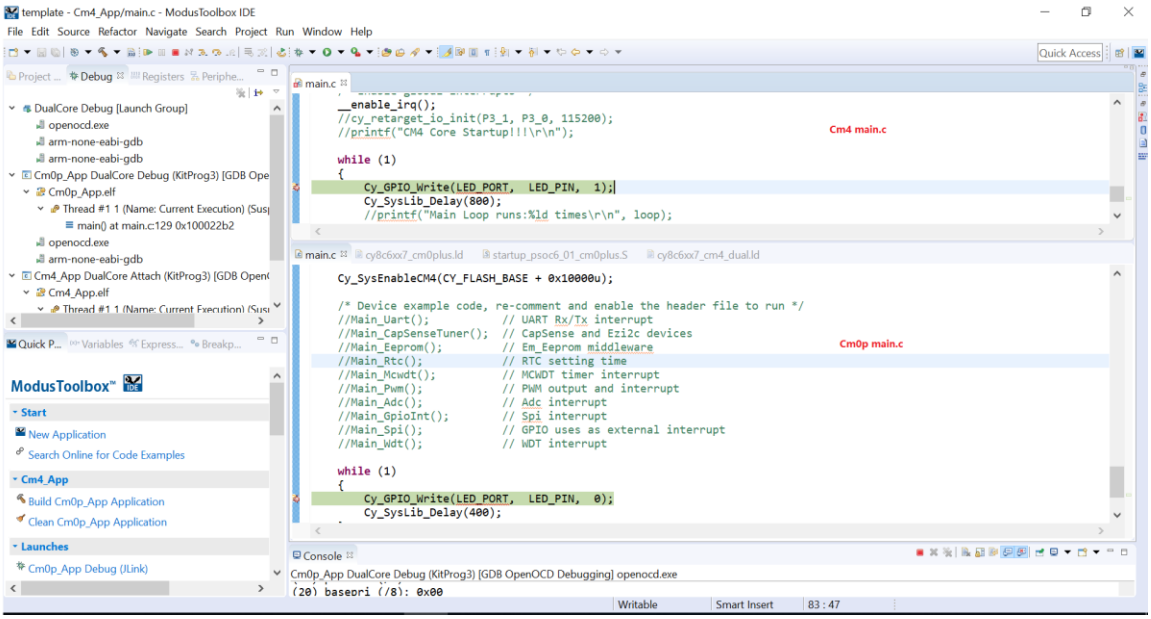
b. Add “Cm0p_App DualCore Debug (KitProg3)” and “Cm4_App DualCore Attach (KitProg3)” items in sequence via “add” button. The click “Apply” button to save the configuration.



c. Selected Cm4_App project, then start the “DualCore Debug (KitProg3)”.



d. Debugging have jumped to Cm0p main(). After running to “Cy_SysEnableCM4(CY_FLASH_BASE + 0x10000u);”, we can debug the projects separately.



Attention

1. The path of MTB project cannot contain “space”.

2. You can adjust the Flash/Ram size of CM0p and CM4 core via the linker file in /libs/TARGET_CY8CKIT-062-BLE/linker/TOOLCHAIN_GCC_ARM/ folder.

3. Make sure the address of following function in your CM0p main.c matches the flash assignment.
   `Cy_SysEnableCM4(CY_FLASH_BASE + 0x10000u);`

4. Before starting the debugging, please select the corresponding project.

Troubleshooting

1. Building error, remind “ ***“region `flash' overflowed by 6360 bytes”*** ”.

A: The flash size for project is not enough, modify the linker file and try again.

2. Building error, remind “***303 cannot move location counter backwards (from 08000bf0 to 08000000***)”.

A: The ram size for project is not enough, modify the linker file and try again.

3. Linking error after building, remind “ ***../xxx/xxx.o: No such file or directory*** ”

A: Be sure the path does not contain any “Space”.

4. Building error, remind " ***'cy_capsense_tuner' undeclared (first use in this function); did you mean 'Main_CapSenseTuner'? "etc.*** when using CapSense on Cm0p core?

A: Two ways to fix this issue, in ../lib/TARGET_XXX/ COMPONENT_BSP_DESIGN_MODUS/ folder

1) Modify the macro **#define** CY_CAPSENSE_CORE 0u) in / GneratedSource /cycfg_peripherals.h file by manual when updating the device configurator every time
2) Open the design.modus by editor then modify: <Param id="CapSenseCore" value="0"/>

Reference

1. LWAX-6: Kaadas MTB 2.0 dual-core on-site support

END OF MEMO