



CMPE-250
Assembly and Embedded Programming
Spring 2021
Laboratory Exercise Five
NXP Freedom Board FRDM-KL05Z Configuration and
Keil MDK-ARM Hardware Debugging Tutorial

This first part of this exercise introduces the CMPE-250 lab kit, which includes the NXP Freedom Development Platform FRDM-KL05Z, (Freedom Board). The objectives of this exercise are to configure and verify proper operation of the NXP FRDM-KL05Z and to learn how to use Keil MDK-ARM for hardware debugging. An assembly language program is downloaded to the FRDM-KL05Z and debugged in Keil MDK-ARM, and various questions explore important features of assembly language programming of hardware with Keil MDK-ARM. This exercise assumes prior familiarity with simulating assembly language programs in the Keil MDK-ARM environment.

Procedure

These steps are intended for a course lab workstation with an NXP FRDM-KL05Z board that has already been configured in the course lab kit. Additional steps may be required for a different version of Windows or a board with a different configuration, as indicated in this document by the  computer icon or the  board icon, respectively.

Verifying the NXP FRDM-KL05Z configuration

1. Inspect the CMPE 250 lab kit to verify it contains the following items.
 - ___ NXP FRDM-KL05Z
 - ___ USB cable: standard-A (host) plug to mini-B (device) plug
2. Verify the operation of the FRDM-KL05Z as Arm Mbed DAPLink for CMSIS-DAP. (These instructions assume the board has the original demonstration program as shipped from NXP and has already been configured with DAPLink for the course lab kit.)
 - a. Insert the mini-B plug of the USB cable into the Open SDA USB connector of the FRDM-KL05Z, which is shown in Figure 1.
 - b. If the USB cable is not already plugged into the lab workstation, insert the standard-A plug of the USB cable into a free USB port of the lab workstation.
 - c. When Windows recognizes the board for the first time, it typically generates several notifications at the right side of the Windows taskbar.
 - i. First, Windows may give a notification to select what happens with removable drives. Since the board will not be used directly as a removable drive in this course, click the notification to launch the select options window, and then select Take no action, as shown in Figure 2.

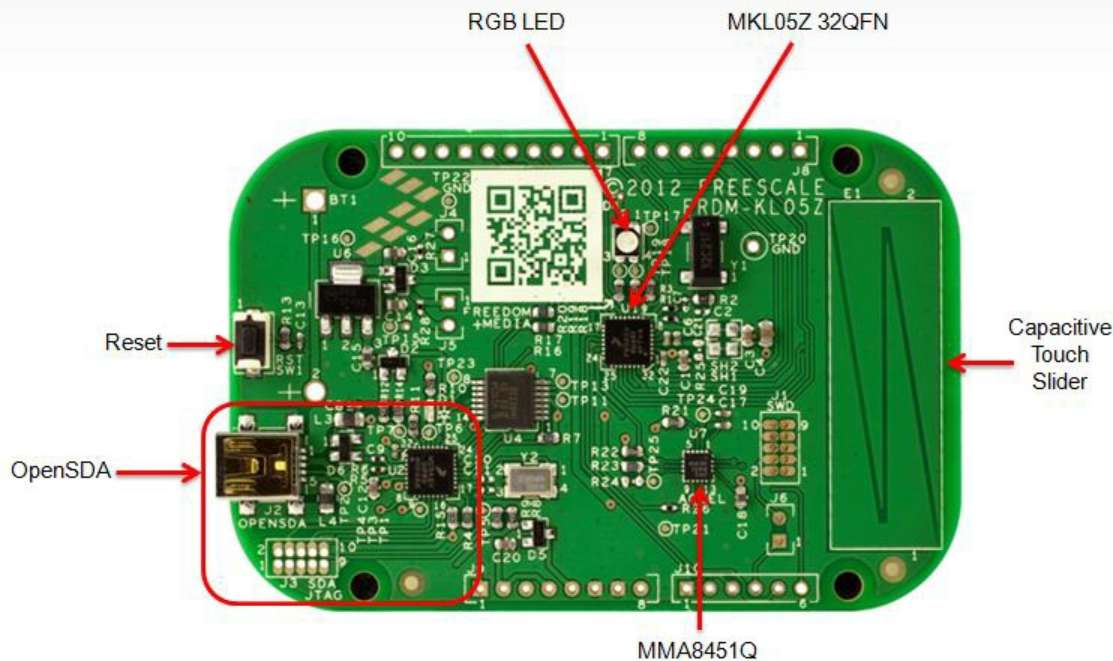


Figure 1. NXP FRDM-KL05Z connectors and components
[from *FRDM-KL05Z User's Manual* (FRDM-KL05Z-UM), Rev. 1.0]

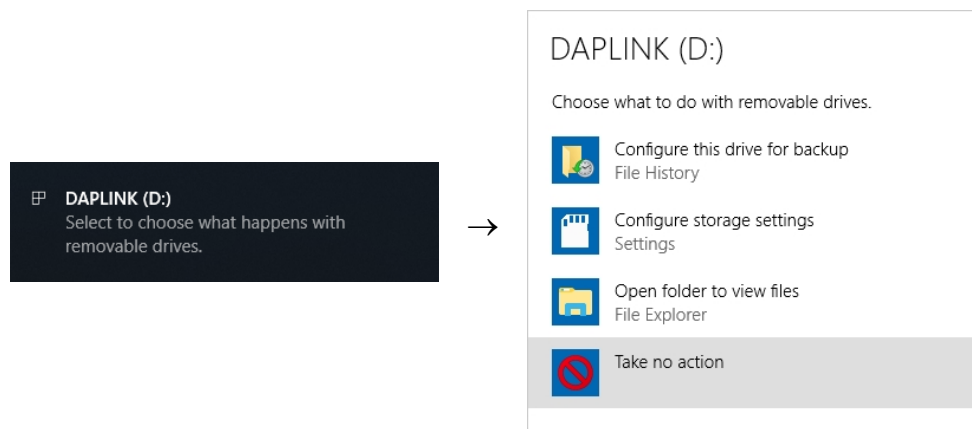


Figure 2. Windows select removable drive options

This tutorial was developed for Keil MDK-ARM Version 5.31.0.0 with Keil device pack Kinetis_KLxx_DFP.1.14.0 and for NXP FRDM-KL05Z with OpenSDA ARM Mbed DAPLink v2.53.

NXP, Freedom Development Platform, and Kinetis® are trademarks or registered trademarks of NXP Semiconductors. OpenSDA is an open-standard serial and debug adapter built into several NXP evaluation boards.

Arm® and Mbed™ are trademarks or registered trademarks of Arm Limited (or its subsidiaries). Arm Mbed DAPLink is an open-source software project.

Microsoft® and Windows™ are trademarks or registered trademarks of Microsoft Corporation.

- ii. Next, Windows should generate a notification that it is setting up a device. It is not necessary to do anything, but clicking the notification will launch the Windows device settings window. After setup, DAPLink CMSIS-DAP will appear under Other devices without any error messages, as shown in Figure 3.
- Windows device setup for the FRDM-KL05Z includes several device drivers (see Figure 6).
 - Disk drive: MBED VFS USB Device
 - Portable Device: DAPLINK
 - Port: USB Serial Device (COM n)
- On some Windows computers outside of lab, Windows may require a restart to complete device setup.
- On some Windows computers outside of lab, DAPLink CMSIS-DAP may not appear under Other devices; however, it will appear in Windows Device Manager under Portable Devices as DAPLINK (see Figure 6).
- iii. Finally, Windows should give a notification that the device is ready, as shown in Figure 4.

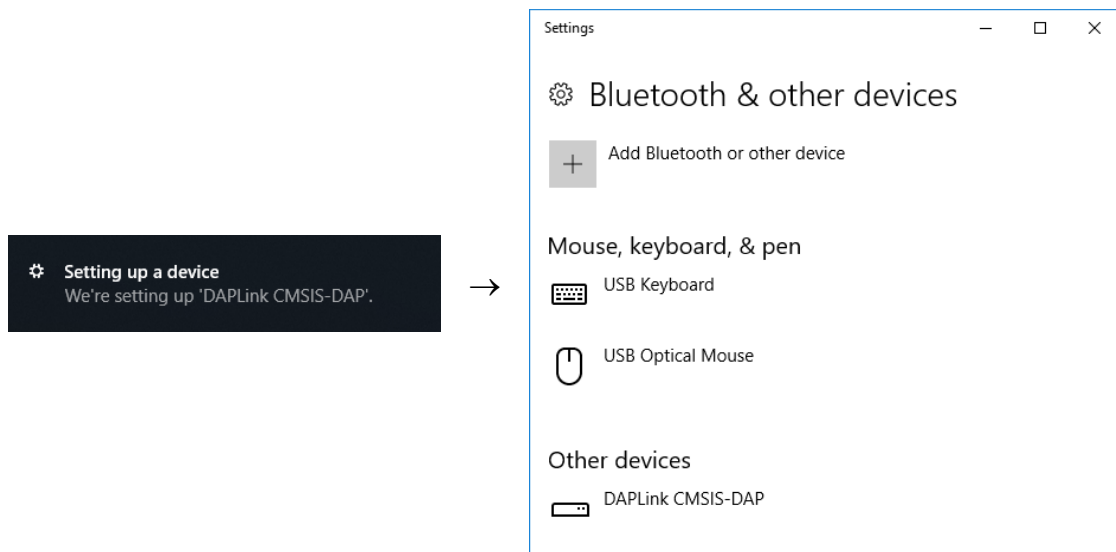


Figure 3. Windows device settings

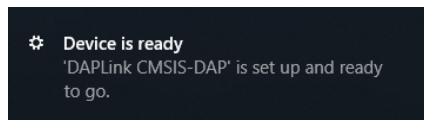


Figure 4. Windows device ready notification

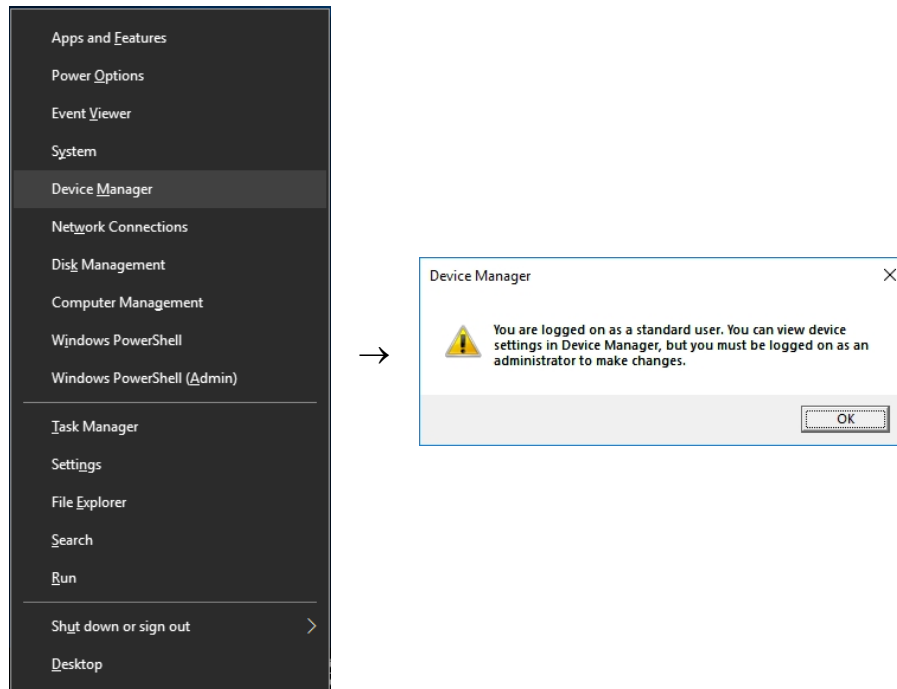


Figure 5. Launching Windows Device Manager

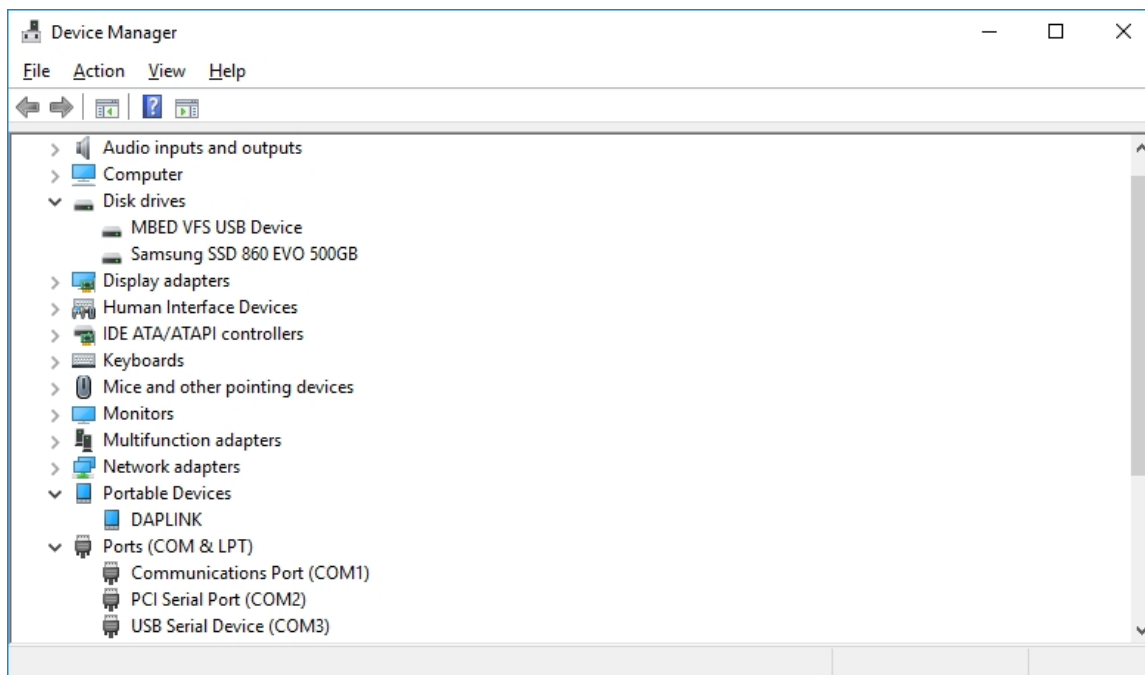




Figure 6. Windows 10 Device Manager showing FRDM-KL05Z drivers

- d. Find the board's COM port number in Windows Device Manager.
 - i. Simultaneously press the  Windows and X keys to launch the Windows Quick Link menu.
 - ii. From the Quick Link menu, select Device Manager, as shown in Figure 5.
 - iii. If a warning dialog appears about being logged on as a standard user, as shown in Figure 5, press the OK button to continue to Device Manager.
 - iv. In the Device Manager window, ∨ expand Ports (COM & LPT) to find USB Serial Device, as shown in Figure 6.
 - v. Note the COM port number for USB Serial Device, and record it below for use later. (For example, in Figure 6 the board has been assigned COM3).

COM_____

- ☞ The COM (communications) port number is assigned during the device setup and will remain the same on this workstation. On a different workstation, device setup may occur the first time the board is connected, and a COM port number will be assigned for that individual board on that workstation. In other cases, Windows may use an existing COM port number associated with a previous DAPLink CMSIS-DAP installation.
 - ☞ Figure 6 shows all of the drivers installed for the NXP FRDM-KL05Z board: MBED VFS USB Device (under disk drives); DAPLINK (under portable devices); and USB Serial Device (under ports).
 - 💡 If the board does not appear in Device Manager as indicated in Figure 6, get assistance from lab staff.
 - ⇒ The FRDM-KL05Z board ships from NXP with a bootloader and firmware that are compatible only with Windows 7, Vista, and XP. For the lab kit, the board has been configured with a bootloader and firmware that are compatible with later versions of Windows. Instructions and files for updating boards not from the lab kit are in myCourses content under Laboratory Exercises → Lab Resources → OpenSDA Update for Windows 10.
 - 💻 Although the bootloader and firmware configured on the board for the lab kit are compatible with Linux and MacOS, the software used to program and debug the board in this course works only under Windows.
- e. At this point on a brand new FRDM-KL05Z board, a demonstration program may be running. The pre-installed demonstration program first cycles through several colors on the RGB LED. After this completes, tilting the board will change the color mix of the RGB LED, based on accelerometer data, and adjusting the position touched on the capacitive touch slider will determine the brightness of the LED.
 - ☞ This demonstration program will be removed from the board in the procedure that follows. If it is wanted at a later time, it can be obtained from the FRDM-KL05Z Quick Start Package available from the NXP web site.

Configuring Keil MDK-ARM for OpenSDA and ARM DAPLink firmware.

3. Create a Keil MDK-ARM project.
 - a. Just as for previous exercises, create new directory (folder) and Keil MDK-ARM project for this exercise.
 - b. Create the Exercise05.s assembly program file to implement the program specified for Lab Exercise Five, and add it to the project, as in previous exercises.
 - c. Download the Start.s assembly program file to your project directory from myCourses content under Laboratory Exercises → Lab Resources, and add it to the project.
 - ☞ The file from myCourses is a .txt text file, and the type will need to be changed to a .s assembler source file.
 - ☞ When adding an existing file to a project, the Add Files to Group 'Source Group 1' dialog will not close after pressing the Add button, to allow for selecting and adding additional files to the project. To close the dialog, press the Close button.
 - d. Download the architecture MKL05Z4.s file to your project directory from myCourses content under Laboratory Exercises → Lab Resources, *but do not add it to the project.*
 - ☞ The file from myCourses is set as a .txt text file, and the type will need to be changed to a .s assembler source file.
 - ☞ This hardware file contains various EQUates needed for the main program source file and for Start.s, but it does not get compiled to a separate object file.
4. Set the target options for mbed OpenSDA and CMSIS-DAP.
 - ☞ Note: the target options set in the steps below are *in addition to* the usual project target options set for previous exercises—make sure they are set as well.
 - a. On the second toolbar row, click the  Target Options icon (or select menu Project→Options for Target 'Target1' ...) to launch the Options for Target 'Target1' window, as shown in Figure 7.

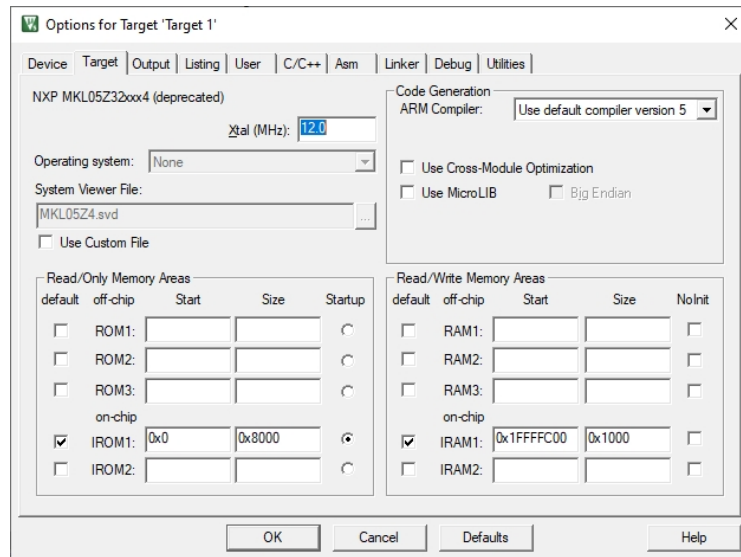


Figure 7. Options for target microcontroller

- b. Click on the Listing tab along the top of the window to get the options shown in Figure 8.
- c. In addition to setting Page Width to 120 and Page Length to 49 as in previous projects, in the Assembler Listing section below them, uncheck the Cross Reference option, as shown in Figure 8.
 - ☞ Leaving the Cross Reference option checked would produce a very large listing file since it would contain every EQUated symbol from the architecture MKL05Z4.s file.

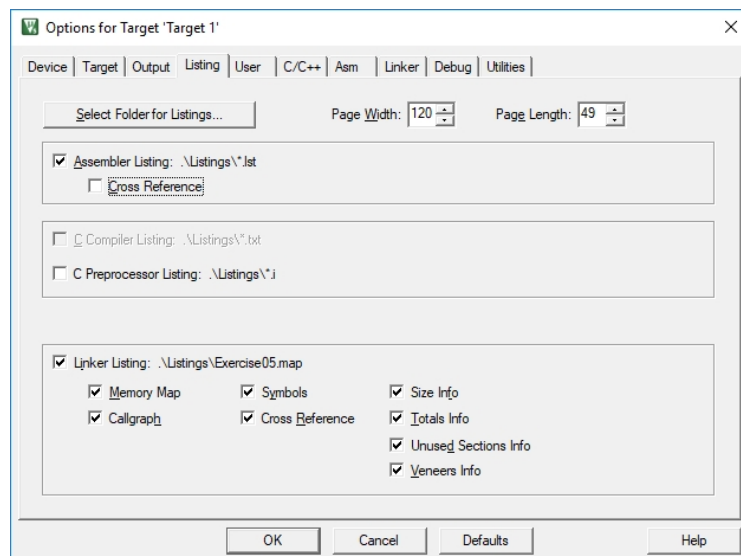
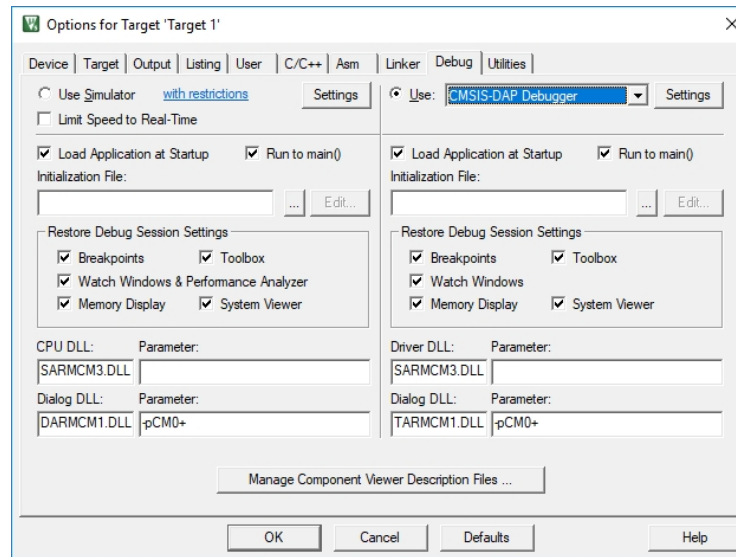
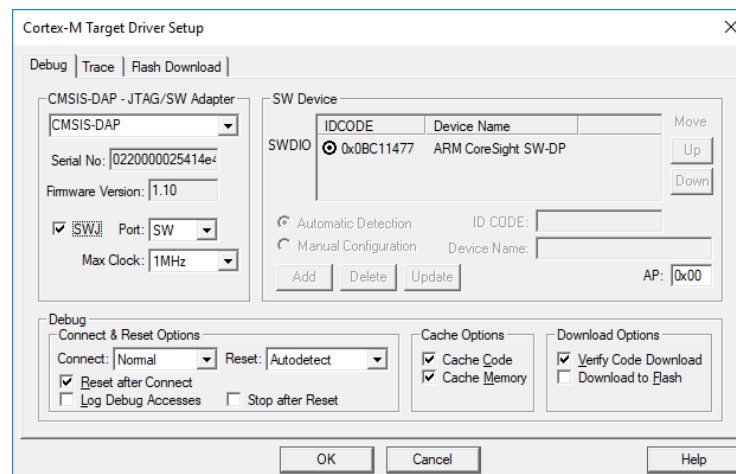


Figure 8. Options for listing output

- d. Click on the Debug tab along the top of the window to get the options shown in Figure 9.

Figure 9. Options for μ Vision debug

- e. At the top right, under the tabs, use the pull-down dialog to the left of the Settings button to select CMSIS-DAP Debugger, as shown in Figure 9.
- f. Select Use to the left of CMSIS-DAP Debugger, as shown in Figure 9.
- g. Click the Settings button to the right of CMSIS-DAP Debugger to launch the Cortex-M Target Driver Setup window, shown in Figure 10.

Figure 10. Debug options for μ Vision Debug

- h. In the CMSIS-DAP – JTAG/SW Adapter section at the left of the window, verify the following settings, making changes if necessary, as shown in Figure 10.
 - i. Make sure CMSIS-DAP is selected in the pull-down menu.
 - ii. Make sure SWJ is checked.
 - iii. Make sure SW is selected for Port.
- i. In the SW Device section at the right of the window, ARM CoreSight SW-DP should automatically be selected to confirm a connection to the target processor on the FRDM-KL05Z. If it is not automatically selected, ask lab staff for help.

- j. Click the **Flash Download** tab to show the flash download options, as in Figure 11.

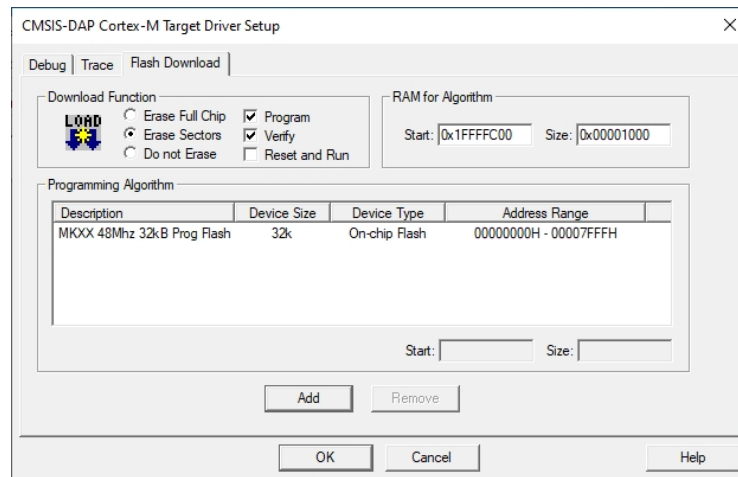



Figure 11. Flash download options for μ Vision Debug

- k. Verify that the settings in the window are the same as those shown in Figure 11.
l. Click OK to return to the target options window.
m. Click OK to return to the main μ Vision window.

Running on the NXP FRDM-KL05 Board with the Keil MDK-ARM Debugger

5. Create an executable.

- On the second toolbar row, click the  Build icon (or select menu Project→Build target) to build the project.
- In the Build Output pane at the bottom of the window, verify that the next to last line of text shows 0 Error(s), as appears in Figure 12. If so, the build was successful; otherwise, any error(s) must be corrected before an executable file is produced.

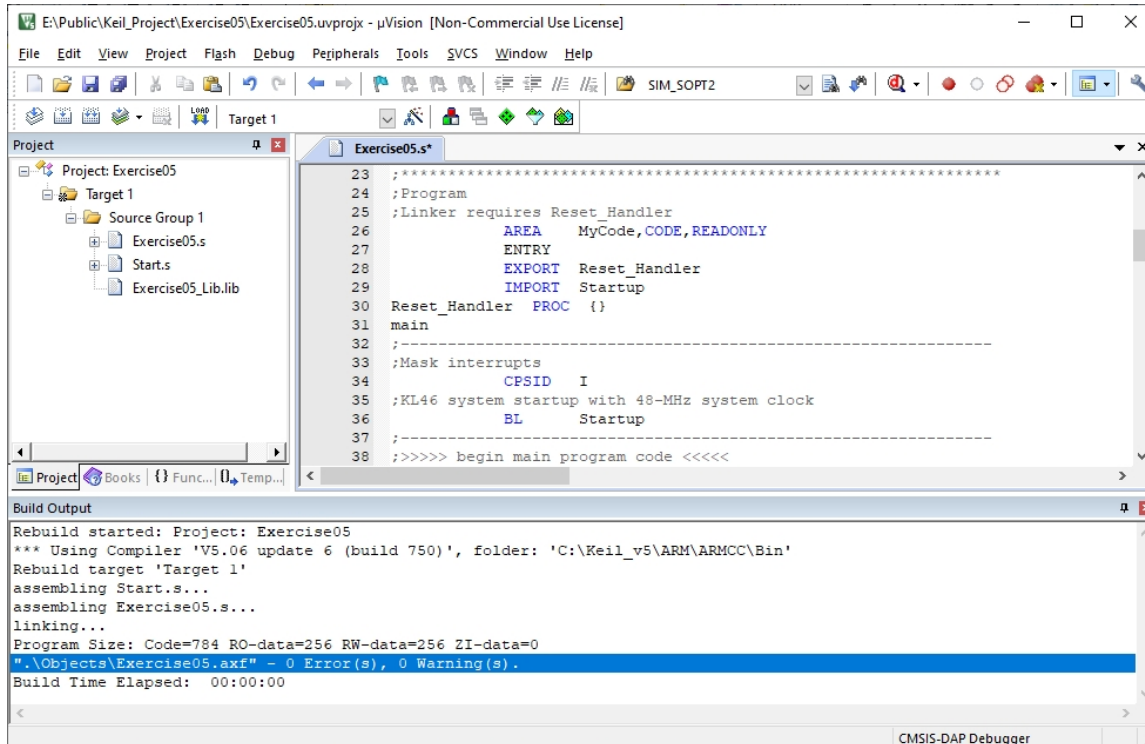





Figure 12. Project after build

- Click the  Save All icon to save all files in the current project.
 Note: This step will preserve all project settings and code so far.
- #### 6. Debug the program with the Keil MDK-ARM Debugger.
- On the top toolbar row toward the right of the window, click the  Start/Stop Debug Session icon (or select menu Debug→ Start/Stop Debug Session) to start a debugging session.
 Note: If you wanted to run the program on the board without using the debugger, you would click the  Download icon to download the executable to the KL05's flash memory, and then you would press the board's reset button RST SW2 to start running the program. Until a different program was downloaded to the board, this program would run whenever the board was powered on or reset.
 Note: Starting the debugger stops the execution of the program.
 Warning: While using the debugger, do not use the board's reset button RST SW2 to start or reset the program; instead use the debugger controls.

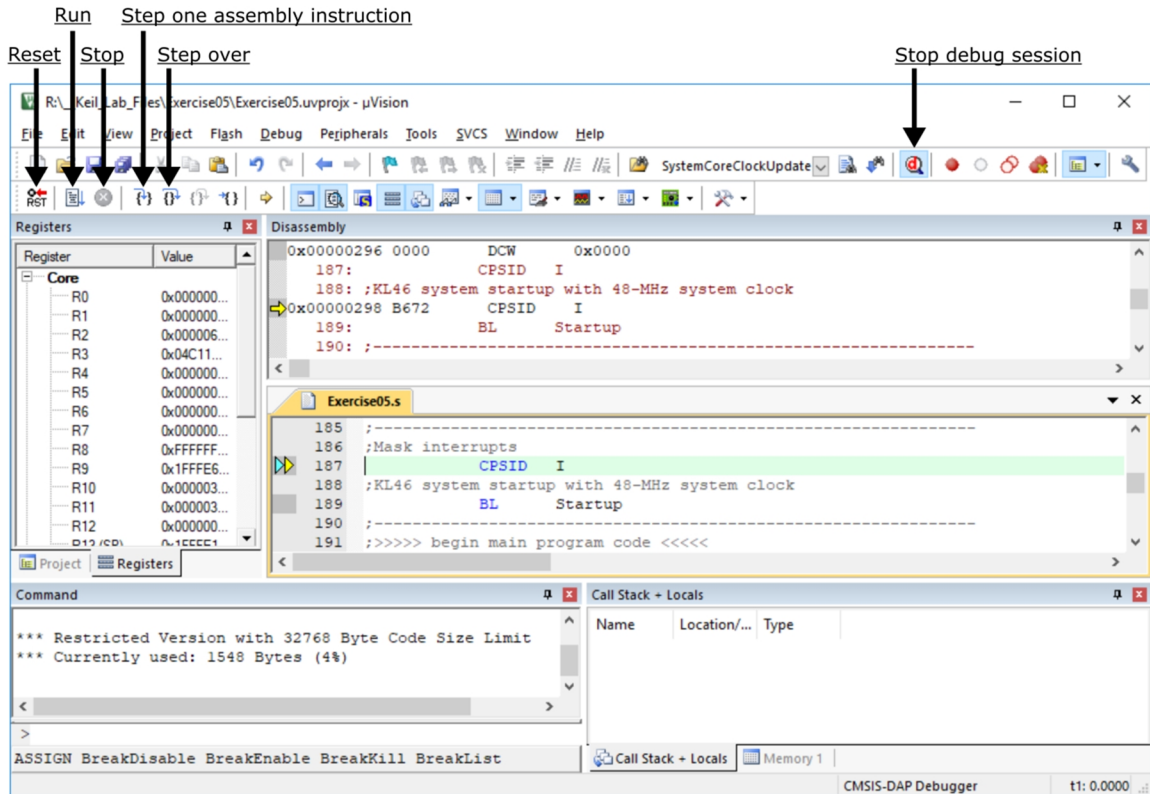


Figure 13. Keil MDK-ARM debugger window starting view

- b. The initial state of the debugger should be similar to what is shown in Figure 13.

Note: Figure 13 shows a clean debug start in the Command pane at the lower left of the debugger window. However, depending on what debug/flash options have been chosen for the project as well as the last state of the debugger, (e.g., from a previous debugging session), the debugger may have error messages in the Command pane at startup, such as “Cannot access Memory.” Such errors at startup are not problems for debugging but rather indicate that a memory viewer or download setting is trying to access memory that is not available at startup. If the associated address viewer or debug settings are changed, the messages will no longer be present when the debugging session starts. Ask lab staff for clarification if there are errors indicated when the debugger starts.

- Q1.** How do the initial contents of the registers differ at the start of this hardware debugging session versus what you have seen at the start of simulator debugger sessions in prior lab exercises?



- Q2.** How do the initial contents of RAM differ at the start of this hardware debugging session versus what you have seen at the start of simulator debugger sessions in prior lab exercises? (Hint: Configure the Memory viewer pane at the lower right of the debugger window to show memory.)



- c. Figure 13 indicates the icons for various debugger commands. The steps that follow explore using these commands to debug the program.
- d. Click the Step icon to execute the `CPSID I` instruction.
- e. Click the Step icon to execute the `BL Startup` subroutine, which will branch to the first instruction of the `Startup` subroutine to step through each of its instructions.
 - ☞ Note: `Startup` is in the `Start.s` source file instead of the main program source file. If the debugger does not automatically open or change context to `Start.s`, you will need to do so manually to trace through the source code.
 - ☞ Note: To execute all of the instructions in the `Startup` subroutine in one step, you would use the Step Over icon, as you will do for `SystemInit` in step 6.g.
- f. Click the Step icon to execute the `PUSH {LR}` instruction.
- g. Click the Step Over icon to execute the `SystemInit` subroutine without having to trace through the execution of each of its instructions.




- Q3.** What registers changed value during `SystemInit`?





- h. Click the Step Over icon to execute the `CPSID I` instruction.
 - ☞ Note: *Step over* a line of source code will execute all of the assembly language instructions that should occur before the next line of source code. For instructions other than `BL`, *step over* is equivalent to *step*.
- i. Click the Step Over icon to execute the `SetClock48MHz` subroutine without having to trace through the execution of each of its instructions.
- j. Find the label for the main loop of your program, left-click the mouse when its pointer is a cursor symbol next to the first instruction of the loop, and press the Insert/Remove Breakpoint icon to set a breakpoint at the instruction.
- k. Click the Run icon to execute all of the instructions from the last executed `BL SetClock48MHz` subroutine up to (but not including) the breakpoint instruction.

- Q4.** Will the program ever end as long as you continue to press the Run icon? Explain why or why not.



1. To close the debugging session, press the  Start/Stop Debug Session icon.
- ⚠ Caution: If a program is actively running in the debugger, it is recommend first to click the  Stop icon to halt execution. The debugger has been known to freeze if a running program is not stopped before attempting to exit the debugger. Following the preceding steps, the  Stop icon in the debugger is grayed out (as it was in Figure 13) at this point since the program is not actively running because execution stopped at the breakpoint.
7. Show your answers to your lab instructor to get your grading credit for the tutorial portion of this exercise.

Setup PuTTY as a terminal for the NXP FRDM-KL05Z

8. Complete the Lab Exercise Five assignment on polled serial I/O.
 - a. Click the  start menu icon at the left of the Windows task bar, and then scroll through the list of program to select  PuTTY. The PuTTY Configuration window will then open, as shown in Figure 14.

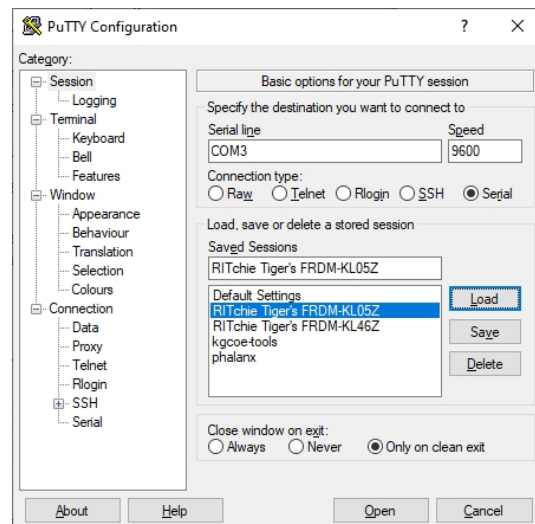



Figure 14. PuTTY Configuration window

- b. On the default initial PuTTY Configuration window, (i.e., Session selected for Category in the left pane), under Connection type in the upper right side of the window, select Serial, as shown in Figure 14.
 - c. Edit the Serial line field just above Connection type to show the COM port number Windows assigned to the FRDM-KL05Z that was noted in step 2.d.v on p. 5, (e.g., COM3 in Figure 14).

- d. (Optional) Save the session settings by typing a name for this session in the **Saved Sessions** in the middle of the window, and click the **Save** button to the right. For example, the settings in Figure 14 have been saved as **RITchie Tiger's FRDM-KL05Z**.
- ☞ When starting PuTTY later with a saved session, rather than repeating the last two steps, you can select the saved session name and then click the **Load** button to the right to restore the correct settings for your FRDM-KL05Z board.
 - 🖥️ If usage history is enabled in Windows, the saved session can be loaded directly when launching PuTTY by right-clicking the  PuTTY icon and then selecting the saved session name from **Recent Sessions** in the pop-up menu, as shown in Figure 15.

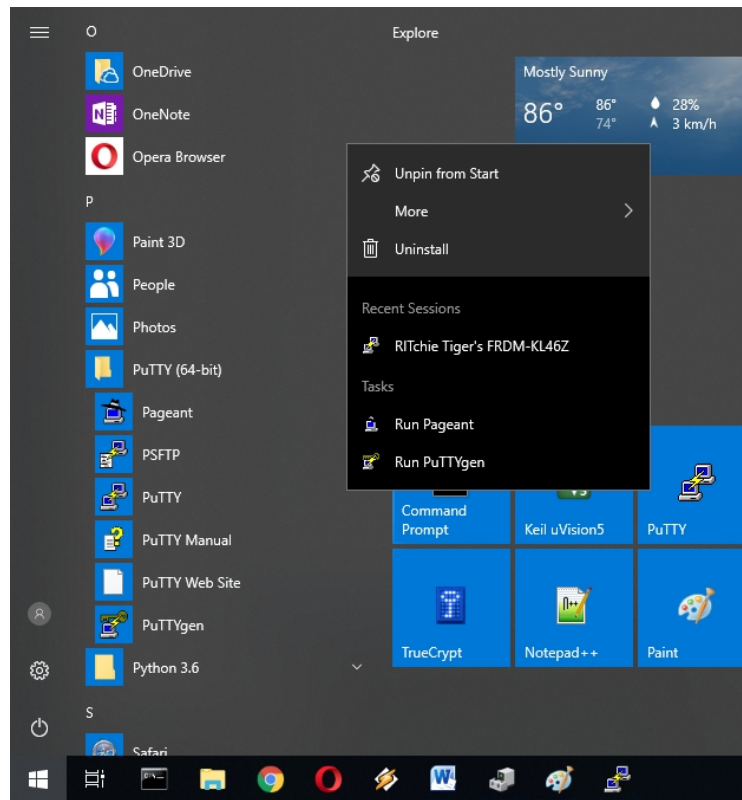


Figure 15. Launching PuTTY with a saved session

- e. On the PuTTY Configuration window, click the Open button at the bottom right to open a terminal window for the FRDM-KL05Z as shown in Figure 16.

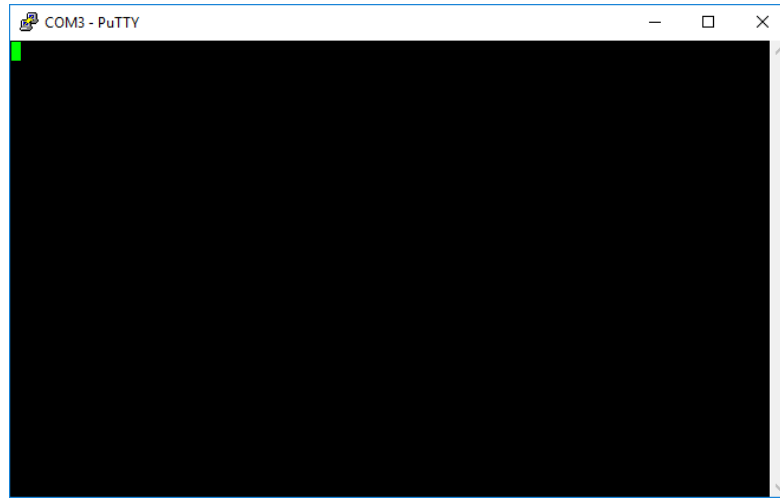


Figure 16. PuTTY terminal window for FRDM-KL05Z on COM3