# STM32 Discovery Board Projects

Notes:

1. This tutorial was written for the STM32F411E-Discovery board and its STM32F411VET microcontroller. If using a different STM Discovery or Nucleo board, basic project setup is the same, except for the microcontroller to be selected.
2. This document assumes that the ARM Keil MDK (Version 5.xx) has been installed, along with the STM32F4xx device family pack and ST-Link USB driver. Refer to the tutorial "Installing MDK-ARM" on the course web page.

## ARM Keil Microcontroller Development Kit (MDK)

As illustrated in Figure 1, MDK-ARM Version 5.xx (***µVision***) comprises a set of core functions:

- *Integrated Development Environment* (IDE) with project manager and editor to create projects
- *ARM C/C++ compiler, assembler, and linker* to build projects (any combination of languages)
- *Debugger* to interactively debug projects on the target board or simulator
- *Pack installer*

One or more software packs must be installed, each of which provides resources (system/startup code, device drivers, etc.) for a specific family of microcontrollers. The appropriate software pack must be installed for each microcontroller family to be used.

- The required pack for these projects is ***Keil::STM32F4xx_DFP***, which supports the ***STM32F411VET*** microcontroller on the ***STM32F411E-Discovery*** board.

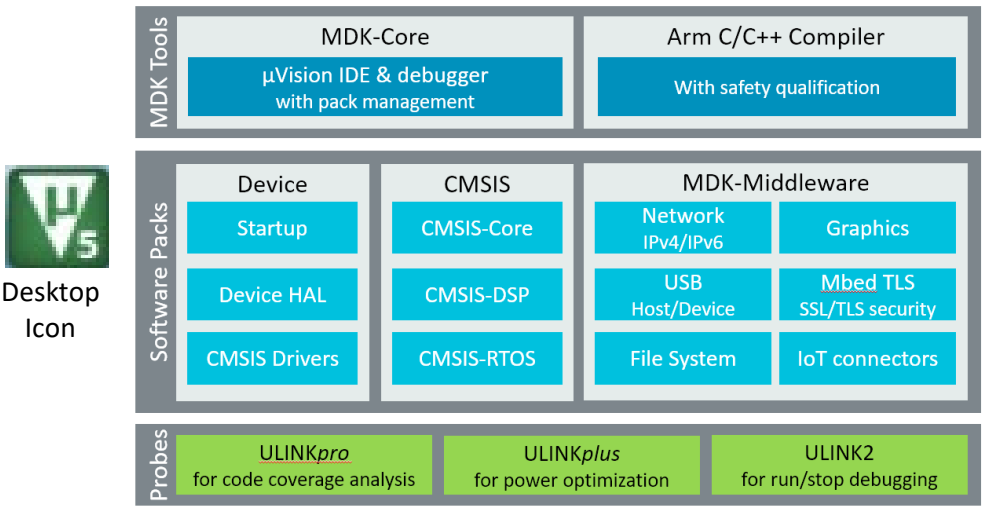



**Figure 1. MDK-ARM V5.xx (*µVision*) comprises the MDK Core and one or more Software Packs**

Creating a New Project for the Discovery/Nucleo Board

1. Open *µVision* and from the menu bar select: ***Project → New µVision Project***. In the Create New Project Window (Figure 1), navigate to the folder in which you want to create the project (create the folder if necessary), enter a project name (ex. *Project1*), and click ***Save***.

   ***It is recommended that a separate folder be created for each project.***

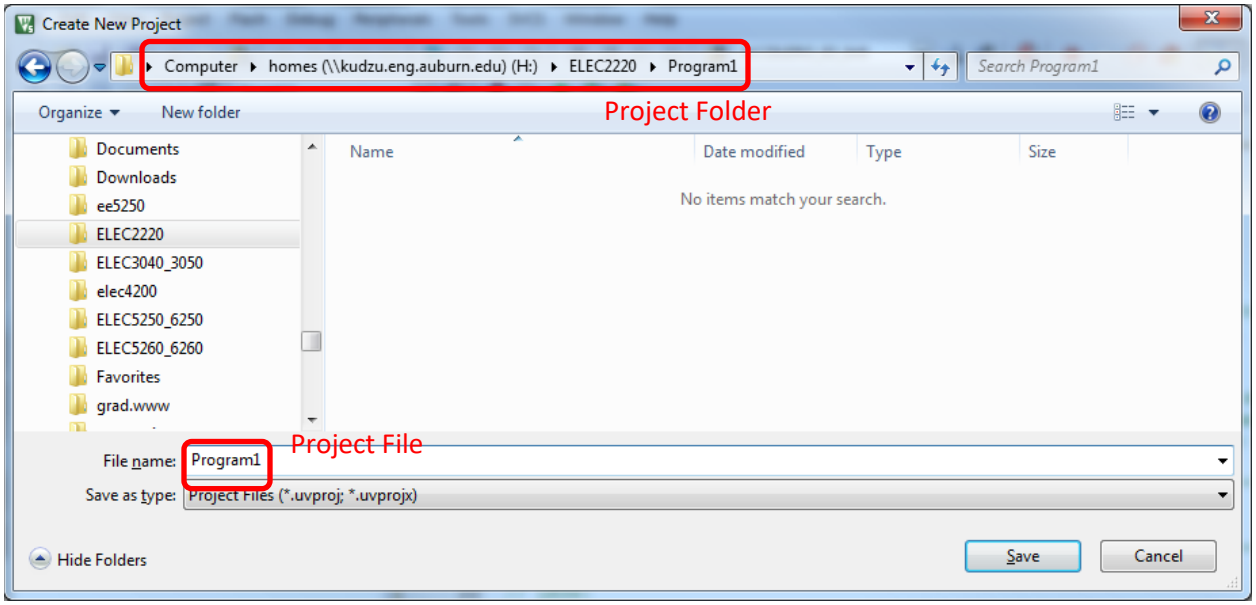



**Figure 1. Creating a new *µVision* project "Lab1"**

2. In the *Select Device for Target* window (Figure 2), select microcontroller ***STM32F411VET***.

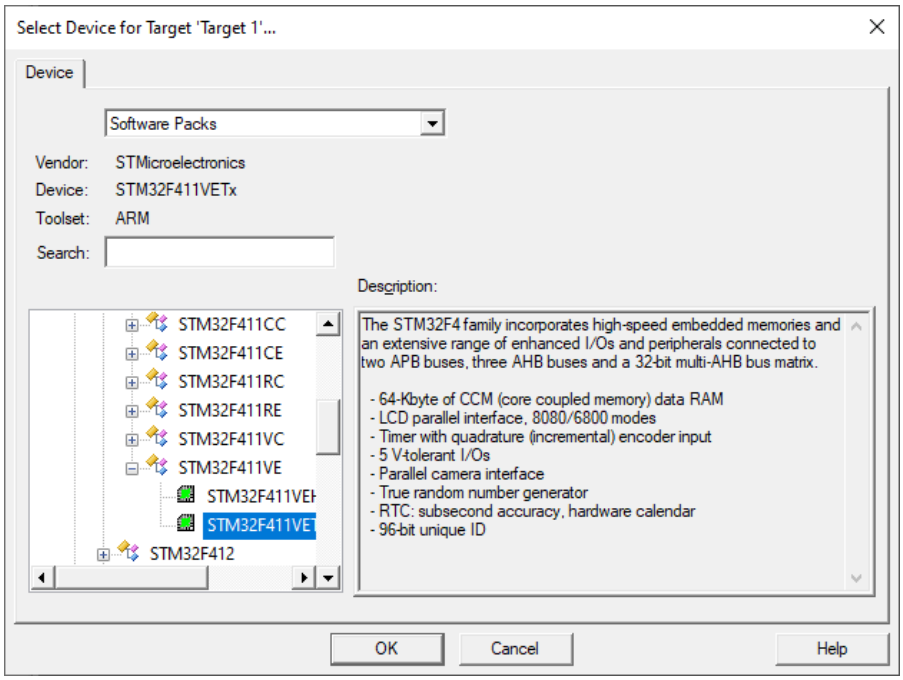



**Figure 2. Select the STM32F411VET microcontroller.**

3. In the *Manage Run-Time Environment* window (Figure 3), you can add microcontroller header and startup files, device drivers, and other software pack components to the project by checking the corresponding boxes. (These can also be added later, if desired.)

## ELEC 2220 - Introductory assembly language programs:

CMSIS Core and Startup programs are unnecessary for testing simple assembly language programs. **Click OK, without checking any boxes,** to omit these from the project. The assembled program code will be placed beginning at the first byte of program memory.

## OTHER PROJECTS:

For other projects: check the boxes for ***CMSIS → CORE*** and ***Device → Startup***. (CMSIS = *Cortex Microcontroller Software Interface Standard*) as shown in Figure 3, plus boxes for any drivers or other software components you wish to include in the project. The assembled/compiled code for your programs will be placed after that of the device startup program in program memory. The startup program is required for initializing the stack, clocks, and some other system resources, after which it jumps to the first instruction of your application program.

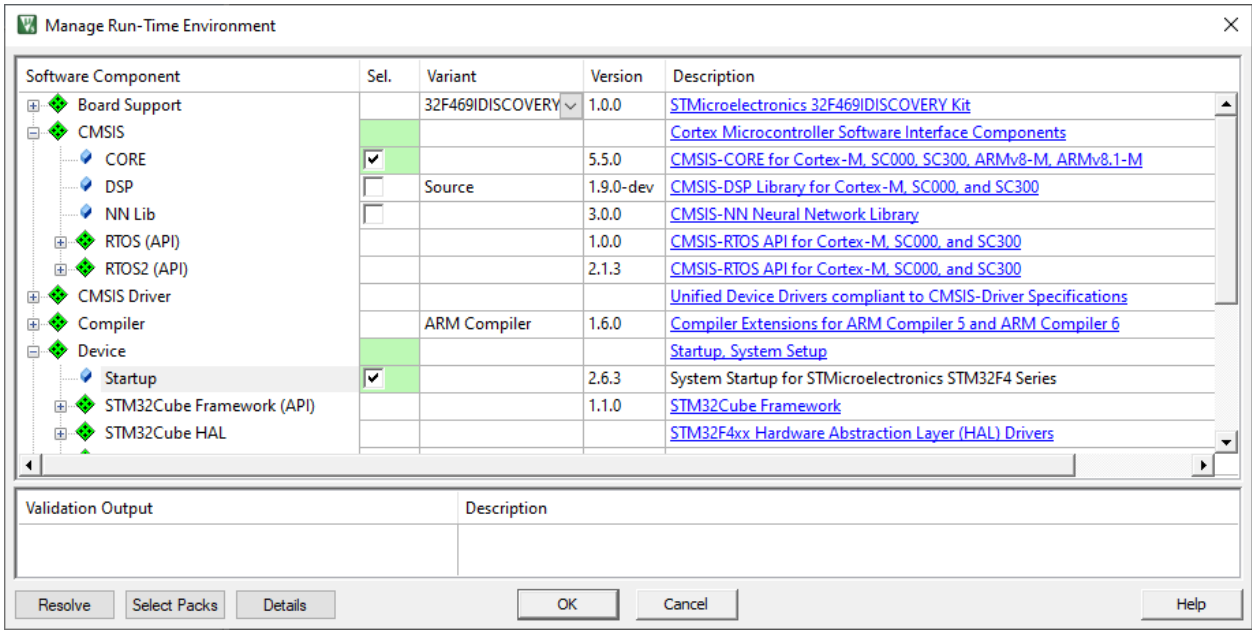



**Figure 3. Include the CMSIS CORE and Startup files in the project.**

**(Unless testing simple assembly language programs in ELEC 2220.)**

4. Create and add source file(s) to the project in the **Project** pane of the *µVision* window (Figure 4). You may do this in either or both of the following ways.
   - Right-click on ***Source Group 1***, select *Add New Item to Group 'Source Group 1'*, select a file type (*Asm File (.s) or C File (.c)*), enter the file name (Example: *Program1.s*), and click ***Add***. This creates a blank file in the editor pane, into which you can type or paste your source program.
   - For files created outside of *µVision*, right-click on ***Source Group 1***, select *Add Existing Files to Group 'Source Group 1'*, locate and select file(s) to be added to the project and click ***Add***.
   - Click ***Close*** after you've created/added all desired source files.

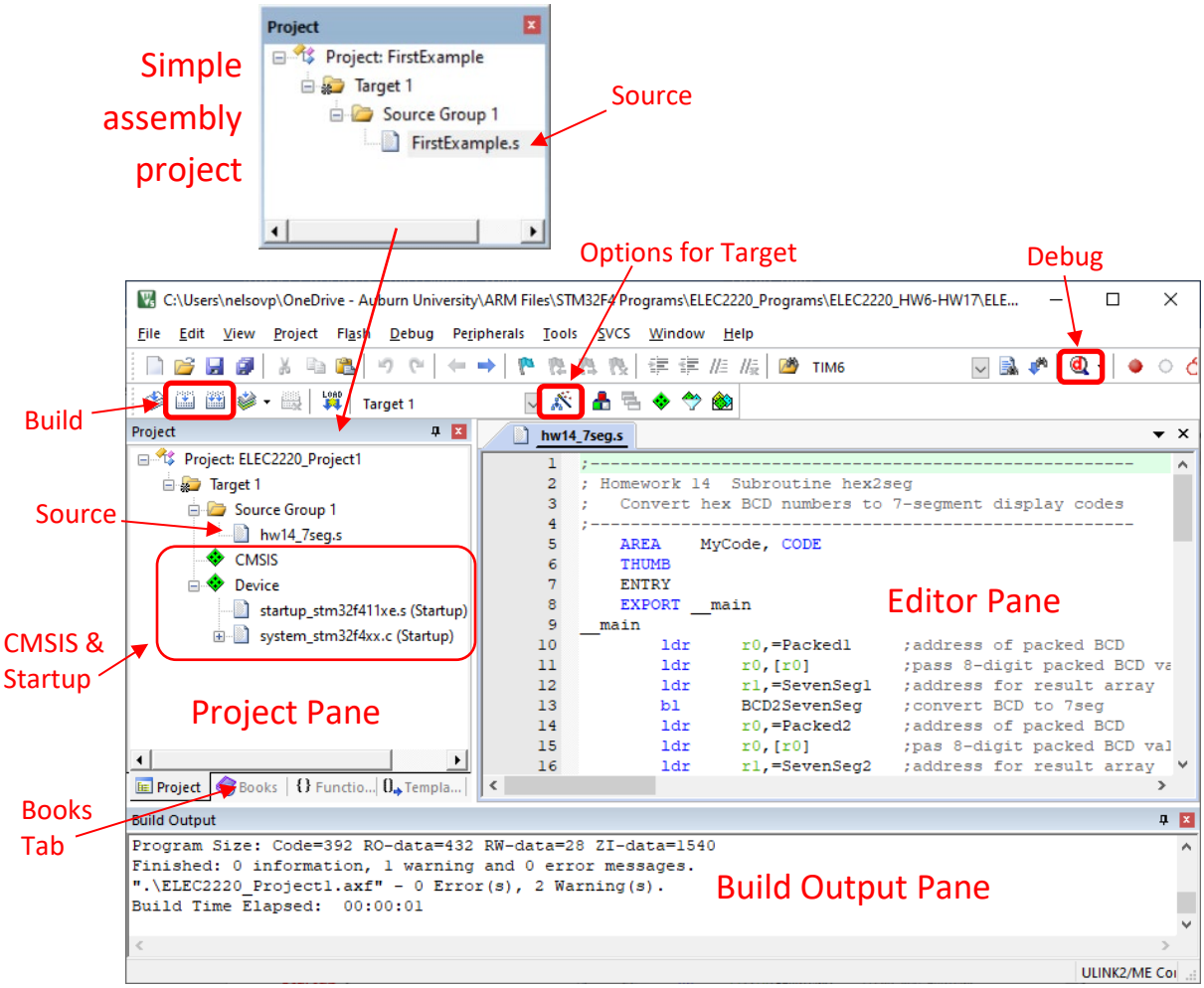
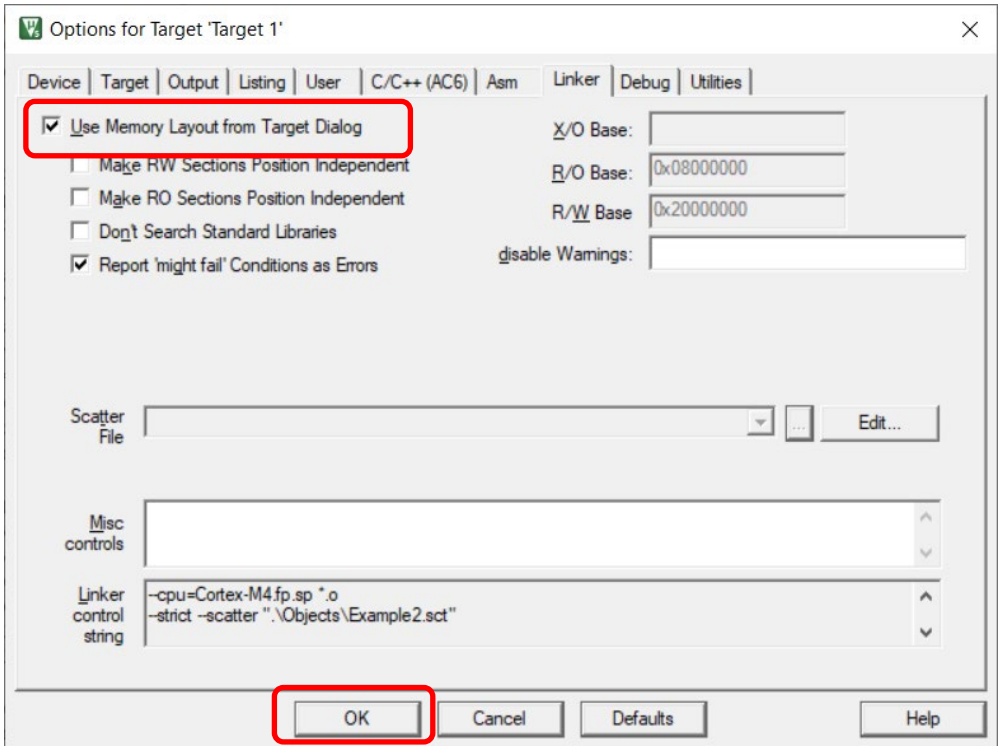



**Figure 4. Add source file(s) to Source Group 1 in the Project pane, and build the project.**
**(Note project pane above for simple assembly projects, without startup/CMSIS/)**

Before going to Step 5, click the "Options for Target" button, and select the Linker tab as shown below. Check the box "Use Memory Layout from Target Dialog" (if not already checked) and click OK to close the window.




5. After creating and saving source files, the project must be "built" by compiling/assembling the source files, correcting any syntax errors as necessary, and then linking the files into one downloadable program. This is performed by:
   - Click on the ***Build*** icon above the project pane (or from the menu bar select ***Project → Build Project***). This will recompile/assemble only source file(s) that have changed.
   - OR: Click on the ***Rebuild*** icon above the project pane (or from the menu bar select ***Project → Rebuild all target files***). This will compile/assemble all source files.
   - Read the messages in the Build Project pane. You must see the message
     ***".\MyProject.axf" - Error(s)", k Warning(s)***
     before you can debug the program. (*MyProject.axf* is the object file (binary machine code) to be programmed into the microcontroller. **If there are any errors, this file will NOT be created, and attempting to enter the debugger will fail.)**
   - If there are one or more errors/warnings, scroll up through the messages in the Build Project pane to identify the reason for each and where they are located in your program. Any errors must be corrected before the object file will be created.

**NOTE:** The project must be rebuilt if you change any device, target, compiler, assembler, or linker options.

6. There are several options for testing and debugging projects with the *µVision* debug tools.
   a) Load the program into the **simulator** for testing, rather than the physical microcontroller *(See to Step 7 below.)*
   b) Download the program to the **Discovery board**, program it into the on-chip flash memory of the microcontroller and execute the program in the target hardware. *(See Step 8 below.)*
   c) Execute a program that was **previously programmed** into the flash memory of the target microcontroller. *(See Step 9 below.)*

7. To download the compiled program to the simulated microcontroller's memory and initiate a debug session with the simulator, set up the project Debug options as follows.
   - From the menu bar, select ***Project → Options for Target 'Target 1'*** (or click on the ***Options for Target*** icon next to the target name in the menu bar.)
   - Select the ***Debug*** tab (Figure 5) and check the ***Use Simulator*** button.
   - If you wish to use a debug initialization file (described in Step 10 below), enter the file name in the ***Initialization File*** box below the ***Use Simulator*** button.
   - Click OK to close the Options for Target window.

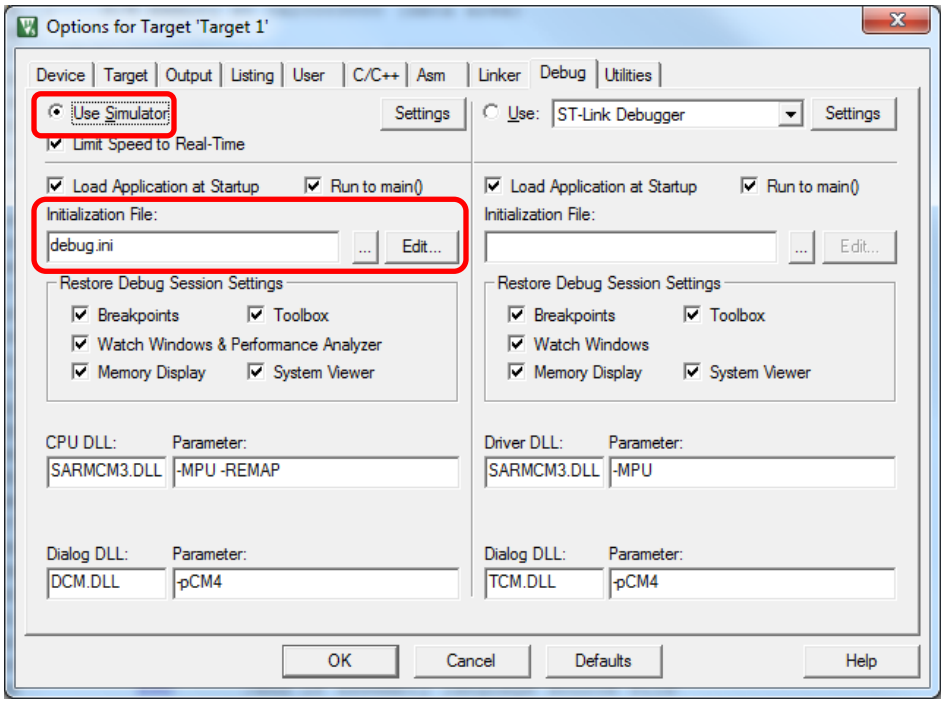



**Figure 5. Target Options Debug tab. "Use Simulator" selected, and debug initialization file specified.**

8. To download the compiled program to the Discovery board, program it into the microcontroller's flash memory, and initiate a debug session, set up the project Debug options as follows.
   - From the menu bar, select ***Project → Options for Target 'Target 1'*** (or click on the ***Options for Target*** icon next to the target name in the menu bar.)
   - Select the ***Debug*** tab (Figure 6), check the ***Use*** button and select ***ST-Link Debugger***.
   - If you wish to use a debug initialization file (described in Step 10 below), enter the file name in the ***Initialization File*** box below the ***Use ST-Link Debugger*** button.

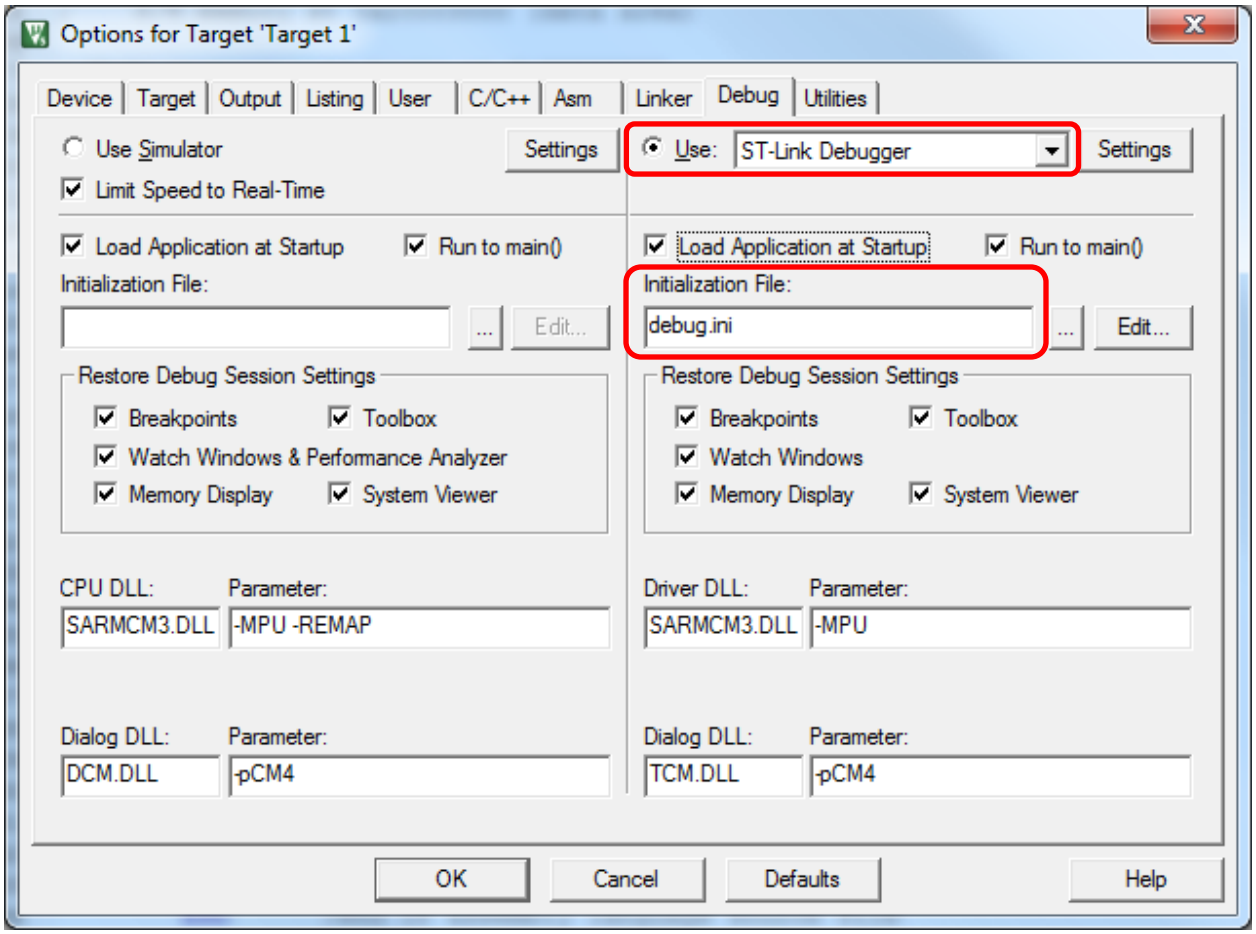



**Figure 6. Target Options Debug tab. "Use ST-Link Debugger" selected.**

**(Continue on next page for setting up the ST-Link Debugger)**

- Click on ***Settings*** next to ***ST-Link Debugger*** to produce the Cortex-M Target Driver Setup Window (Figure 7), and select ***Port SW*** (Single-Wire debug) under ***Target Com***.
  ***If Serial Number is blank, install/reinstall the ST-Link USB driver on your computer.***

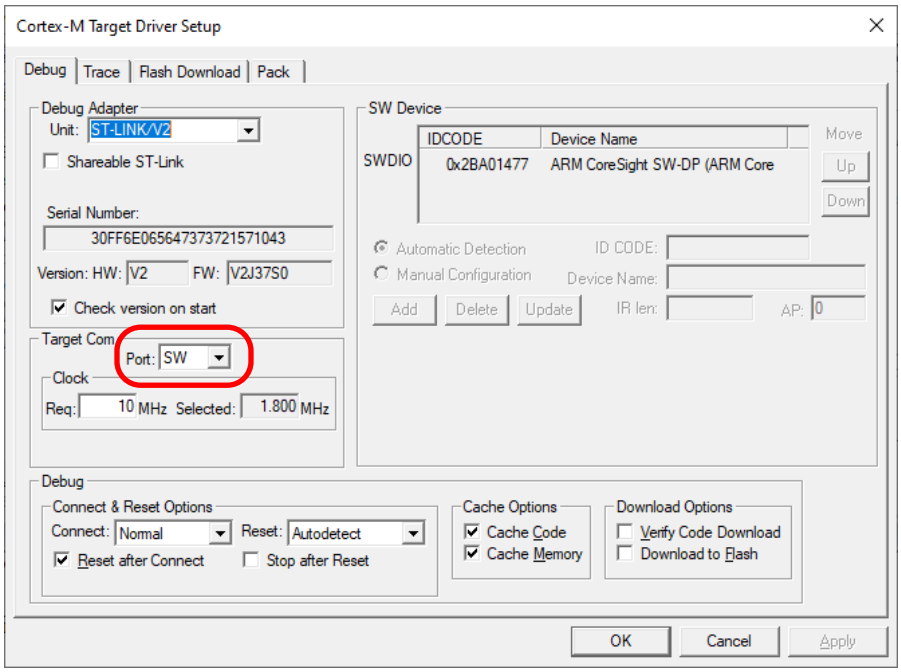



**Figure 7. Single-Wire (SW) Debug Adapter port selected.**

- Click on the ***Flash Download*** tab of the Cortex-M Target Driver Setup Window (Figure 8) and under ***Programming Algorithm***, click the ***ADD*** button and select ***STM32F4xx 512kB Flash***. Then you can close the setup and options windows.

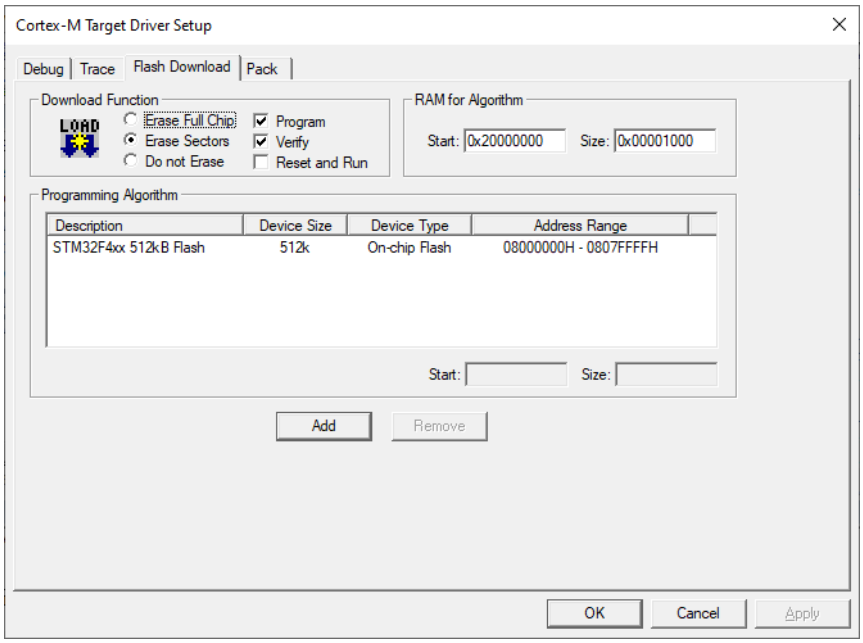



**Figure 8. STM32F4xx flash programming algorithm selected**

9. To initiate a debug session for a program previously programmed into the microcontroller's flash memory on the Discovery board, set up the project Debug options as in step 8 above, but on the Debug tab (Figure 6) uncheck the button ***Load Application at Startup***, located beneath the ***Use ST-Link Debugger*** button. This will cause the debugger to be entered without downloading the program, thereby using the program currently in the microcontroller's flash memory.

10. A **debug initialization file** is useful to set up any desired initial register and memory values and other debug options when the debugger is first opened. This is a plain text file that contains one or more debugger commands; these commands may also be entered in the Debugger's Command Window (see Figure 9).

    The following is a sample initialization file (*debug.ini* in Figures 5 and 6)

```
xPSR=0x01000000     //Set T bit of xPSR Register to Thumb Mode
PC=0x08000000       //Set starting program counter address
E INT 0x20000000 = 5,23,-100,200,201,-200,0,1000,2000
```

    - The first line initializes the T (Thumb) bit to 1 in the Processor Status Register (xPSR) to force the processor into Thumb mode. The Cortex-M4 processor in the STM32F411 microcontroller supports only Thumb mode instructions, and its T bit is hard-wired to 1. However, the simulator initializes xPSR to all 0's, thereby setting T=0, which selects ARM mode. Attempting to execute a Thumb instruction in ARM mode will trigger an error exception. **Therefore, the user must initialize T=1 at the start of simulation when using the simulator.**
    - The second line initializes the program counter (PC) to the address of the first instruction to be executed. This will normally be the first byte of flash memory at address 0x08000000. **If using the device startup code, the initial PC will be loaded from a vector table and therefore this debug command should be removed.**
    - The third line enters a sequence of 32-bit data values (separated by commas) into consecutive memory words, beginning at address 0x20000000 (the first byte SRAM). The "INT" indicates that the data are to be encoded and written as 32-bit values. Type "CHAR" would indicate 8-bit data. This E command is useful for entering test data for small programs.

11. To initiate a debug session, click the ***Debug*** icon in the *µVision* window (Figure 4), or select ***Debug*** → ***Start/Stop Debug Session.*** Depending on how you set up the debug options, this will open the Debug Window (Figure 9) after:
    - downloading the project code to the simulator memory (if set up via Step 7),
    - downloading the project code to the target board and programming it into the microcontroller's flash memory (if set up via Step 8),
    - skipping the download (if set up via Step 10)

    At this point, you may run/stop/step the program, set breakpoints, select watch variables, etc. Click on the ***Debug*** icon to stop the debug session and return to the *µVision* IDE.

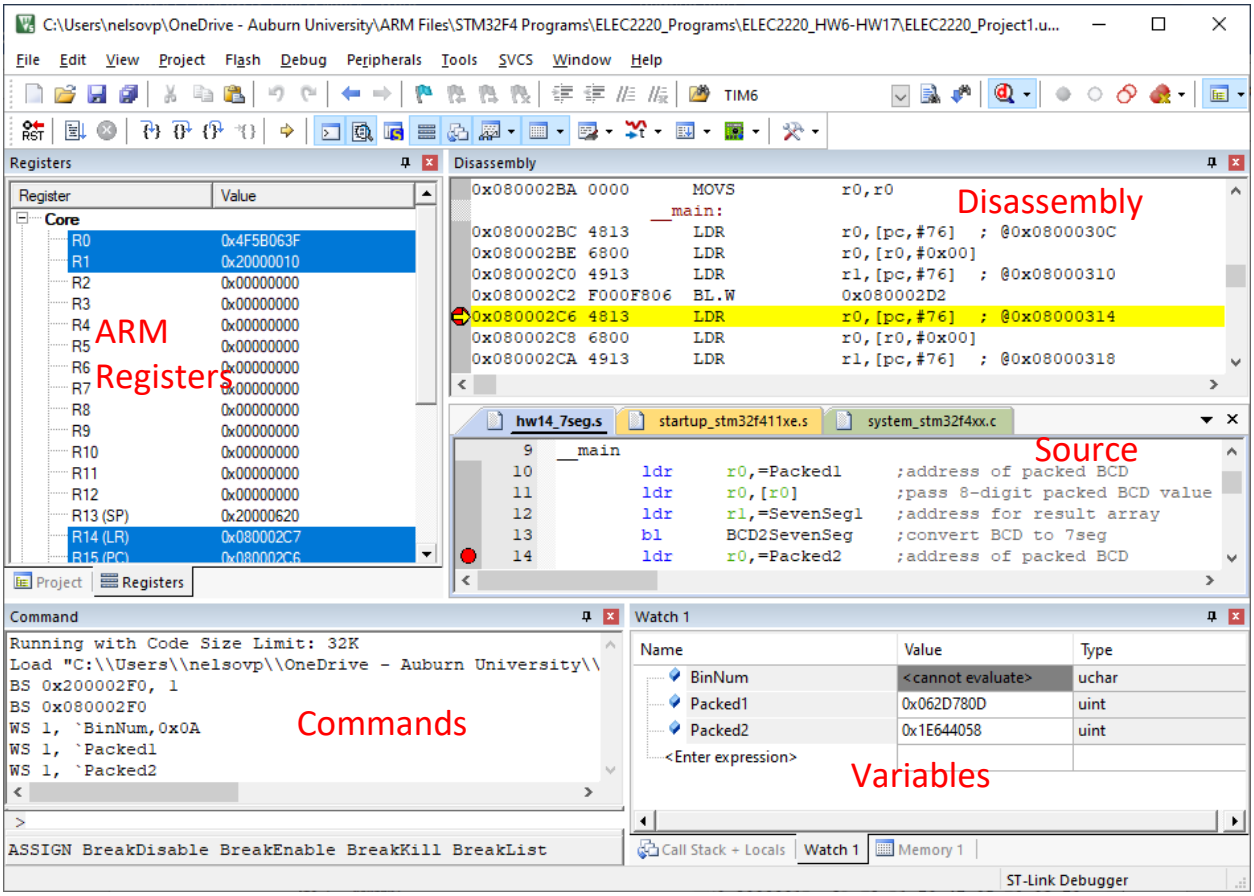



**Figure 9. MDK uVision Debug Window.**

Use of the debugger is described in the document ***Debugging Projects with MDK-ARM***, available on the course web page.

For further information, refer to "*Keil uVision MDK - uVision Lab for the STM32F4 Discovery Board*" (http://www.keil.com/appnotes/docs/apnt_230.asp): App Note 230, Section 20 (Creating your own project from scratch)