**Keil uVision 5 + STM32 Project Setup + Segger J-Link**

**Installation**

1. Download and install Keil uvison 5
2. Use the crack to register it to remove the 32kb limit for free
3. Download and install Keil legacy support from (<http://www2.keil.com/mdk5/legacy/>). This is so that I can use STM SPL instead of the new Software packs that use the newer STM32 Cube HAL

**Creating a New Project + Setting up the environment**

**Important: Right after creating a new project in Keil, it does not automatically write anything to the project file. So if at any step Keil crashes, the project is unreadable\lost. So keep on saving the Keil project often (and right after creation to avoid this)**

1. Download the STM32 SPL for the microcontroller that you are using. Do this by going to the STM website, choosing the microcontroller and downloading SPL from the “Design Resource” section
2. Open Keil + create a new project. Create a new project directory, give a name to the project file and click OK
3. Chose the mcu from the list given by Keil. Note: select “use legacy device” before selecting the mcu
4. Keil will ask you to include the startup file. Click OK. Usually this file should be fine but if you get errors you can always replace this file with the startup file provided by the mcu vendor in their CMSIS package in the path [SPL CMSIS]\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x\Startup\arm\...  
   Here you would need to choose which file to use. It could be hd(high density), md(medium density) etc. Refer to your mcu datasheet to see what density it is or you can see what file Keil included by default and use the file for that same density
5. Categorize the source files in group. Useful and makes code more organized. Create the following groups
   1. Startup (containing the .s startup file)
   2. User
   3. CMSIS
   4. StdPeriph
6. Copy the CMSIS and STM32F10x\_StdPeriph\_Driver folders from the Libraries folder of SPL into your project folder + create a new folder ‘User’
7. Now add the required C files to your project
   1. Add new C file to ‘User’ folder. Name it ‘main’ and append ‘\User’ to Location so that the file gets created in the User folder. The will be our main code file
   2. Add 2 files in ‘CMSIS’ group
      1. Core\_cm3.c – located at CMSIS\CM3\CoreSupport\
      2. System\_stm32f10x.c – located at CMSIS\DeviceSupport\ST\STM32F10x – This file is responsible for setting up the clock, PLL etc. You can use the STM32 clock generator excel tool to generate this file based on your clock settings (not available for all family of mcu. Simpler ones likes STM32f103 don’t have one as they are easy to configure by directly editing system\_stm32f10x.c”
   3. Add files in ‘StdPeriph’ group
      1. Misc.c – located at STM32F10x\_SrdPeriph\_Driver\src – responsible for misc functions. Needs to be included in all projects. Contains peripheral functions for NVIC and SYSTICK.
      2. Peripherals files - located at STM32F10x\_SrdPeriph\_Driver\src – This folder contains files for all the peripherals of the mcu. Add as needed
      3. Stm32f10x\_rcc.c - located at STM32F10x\_SrdPeriph\_Driver\src – this file is needed for Reset and Clock control functions. This should be added to every new project
   4. Stm32f10x\_conf.h – located in [SPL CMSIS]\Proect\STM32F10x\_StfPeriph\_Examples\<Any available project> - copy this file to the ‘STM32F10x\_StdPeriph\_Driver’ folder located in your project folder (copied from SPL). This file is to configure what all peripherals out project would be using (by uncommenting the #define for those peripherals) so that their c/h files can be used in compiling the project
8. Configure compile Options for Target 1 (default target)
   1. Right click on Target 1 and select options
   2. Go to C/C++ tab
   3. In ‘Define’ box add ‘USE\_STDPERIPH\_DRIVER’ this is to ensure STM32 SPL is used
   4. Under ‘Include Paths’ add all the folders in the project directory. Note: This include does not work recursively ie. It does not include subfolders automatically. You need to add each and every folder and subfolder manually.
   5. Go to Output tab
      1. Select folder for objects and select the Objects folder in your project folder
      2. Check ‘Create HEX file’
9. Include stm32f10x.h in your main.c file using #include “stm32f10x.h”
10. Browse to stm32f10x.h in your project folder. Right click and go to settings and uncomment “Read Only”
11. Now in uvision include statement right click and select “Open document stm32f10x.h”. This will open this file.
    1. Here you need to specify the kind of mcu you are using by uncommenting the #define statement in the first couple of lines
    2. Uncomment the line ‘#define USE\_STDPERIPH\_DRIVER’ to use SPL
    3. Verify the HSE oscillator values defined (again In the first couple of lines)
12. Set up Jlink to download code to the mcu + debugging
    1. Go to Target 1 options
    2. Go to Debug tab
       1. In the ‘use’ field, select J-link/J-trace Cortex
       2. Click on Settings beside it
       3. Select Port as JTAG or SW. I will use JTAG
       4. Go to Flash Download tab and make sure correct mcu (with correct density) is listed under Programming Algorithm. If incorrect, remove the existing one and add a new one
       5. Check ‘Reset and Run’. This is automatically reset the mcu whenever a new code is loaded onto it
       6. For more information on setting J-link refer to
          1. [www.keil.com/support/man/docs/link](http://www.keil.com/support/man/docs/link)
          2. [www.keil.com/support/man/docs/jlink/jlink\_configure.htm](http://www.keil.com/support/man/docs/jlink/jlink_configure.htm)
          3. [www.keil.com/support/man/docs/jlink/jlink\_cortexdebug.htm](http://www.keil.com/support/man/docs/jlink/jlink_cortexdebug.htm)

References:

1. <https://www.youtube.com/watch?v=_QXoTNP9GIw>
2. <http://fabioangeletti.altervista.org/blog/set-up-keil-for-stm32-discovery-boards/?doing_wp_cron=1454180905.8481409549713134765625>
3. <http://www.emcu.it/STM32/STM32Library/TwoWordsConcerningSTM32Library.html>
4. <http://www.emcu.it/STM32/STM32VLDiscovery_Test/Things%20to%20remember%20when%20developing%20a%20program%20for%20STM32%20using%20STM32-Library.html>