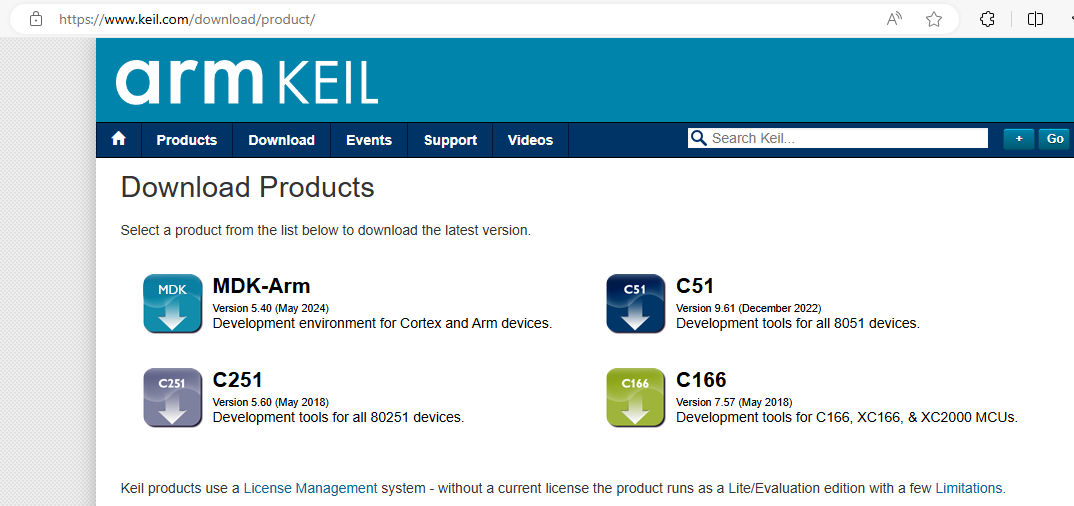
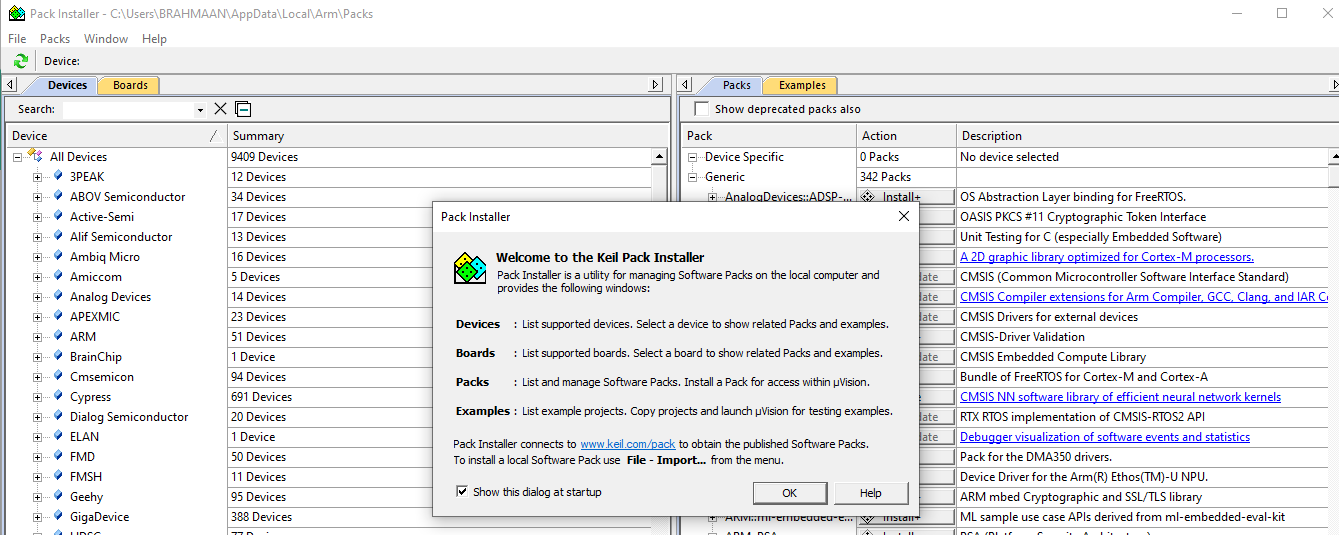
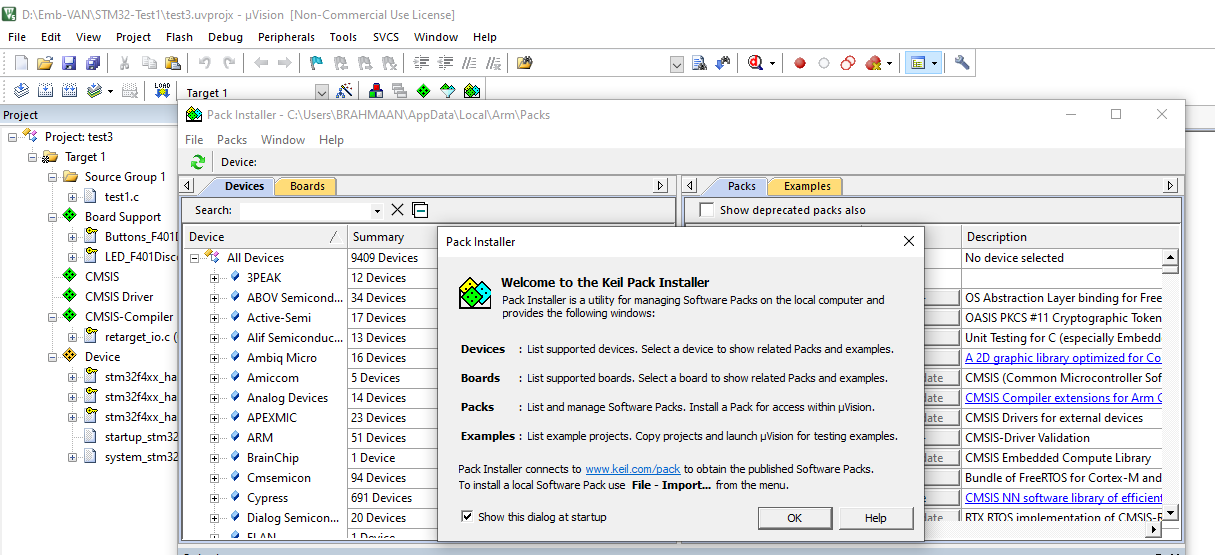
Installation of the Keil uVision Development Software, version 5 (Runs on Microsoft Windows 10 and 11)

1. Go to https:/[/www.keil.com/download/product/](http://www.keil.com/download/product/) page and select MDK-Arm. Download the installer program (a file likely named mdk540.exe) and start it. This is a big file, about 850 megabytes, and during installation other big files will download, so download and install when you have a good internet connection.

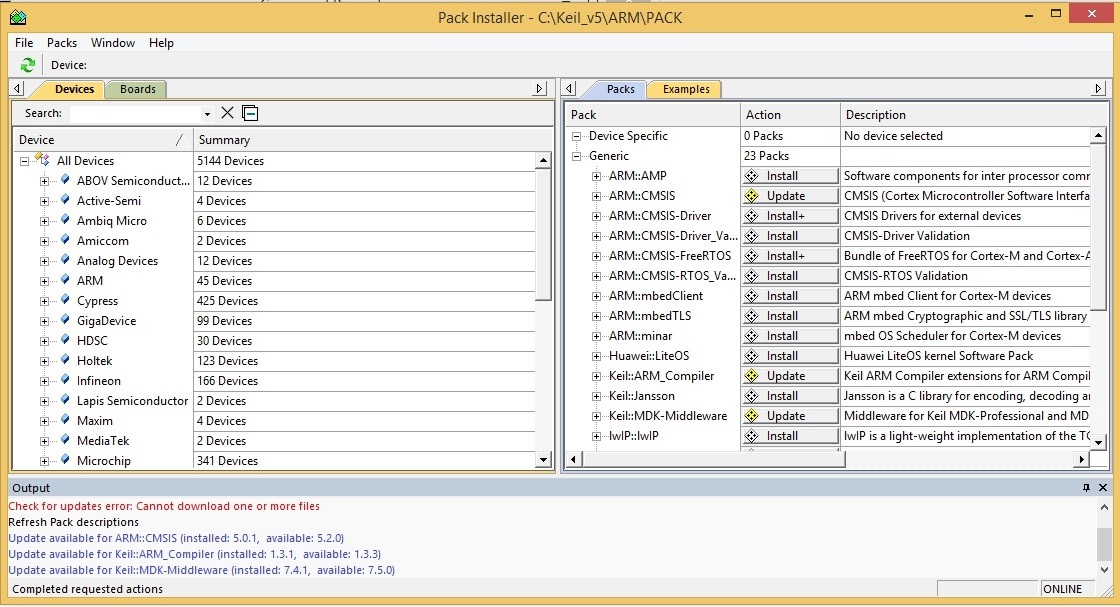


1. You will be asked where to place files. The defaults should be ok.
2. A welcome window will display. Click ok.

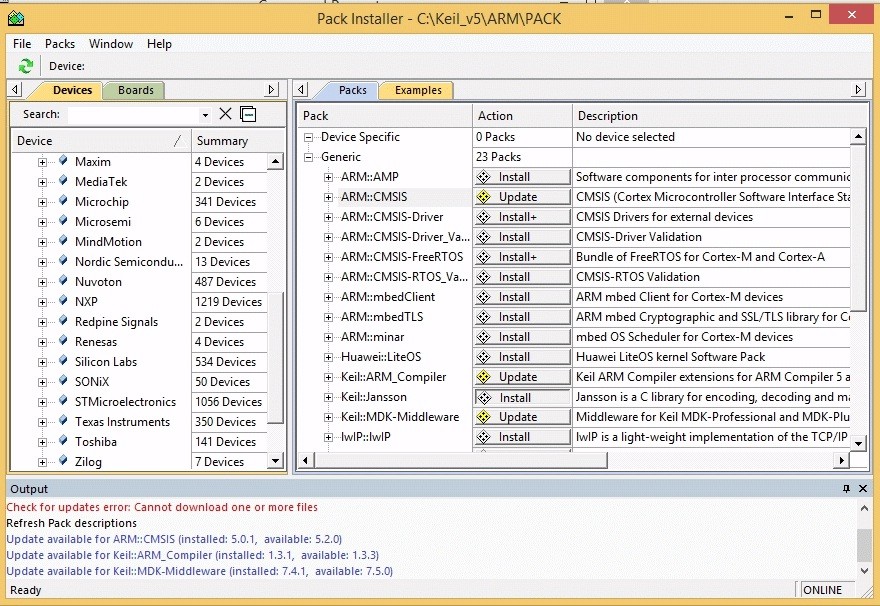




1. The main install window will look something like this. The installer will look for updates. While doing the install for these figures it failed to connect to a third party web site and hence the note (in red) that it could not download one or more files. This will not likely involve software packages that we are concerned with, so you should be able to ignore an error at this point.



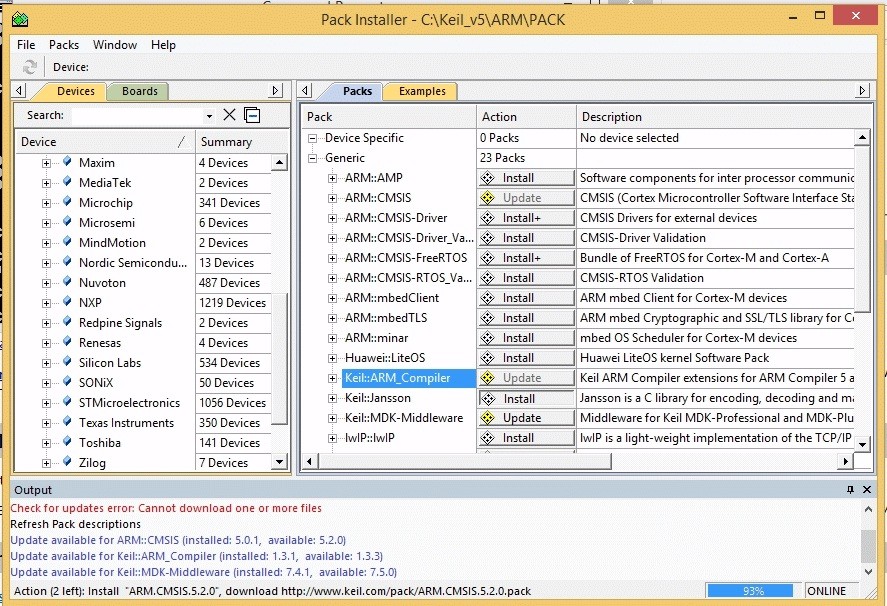
Depending on your screen size etc. you can adjust the width of the various columns of information displayed to allow reading it as well as the window itself.



1. Under Packs in the right hand part of the screen you should click to select Update of the CMSIS core package, Install the Bundle of Free RTOS, and Update the Keil ARM compiler extensions as noted above. The other items will not be needed for our project and most are not useable with the free version of uVision.

Note at lower left that it states Ready and at lower right ONLINE. When you select items to install or update the word Ready will change to a phrase showing what is loading or is selected to load. To the right near the ONLINE word will be a progress indicator.

After selecting items to update/install here is an example screen shot:



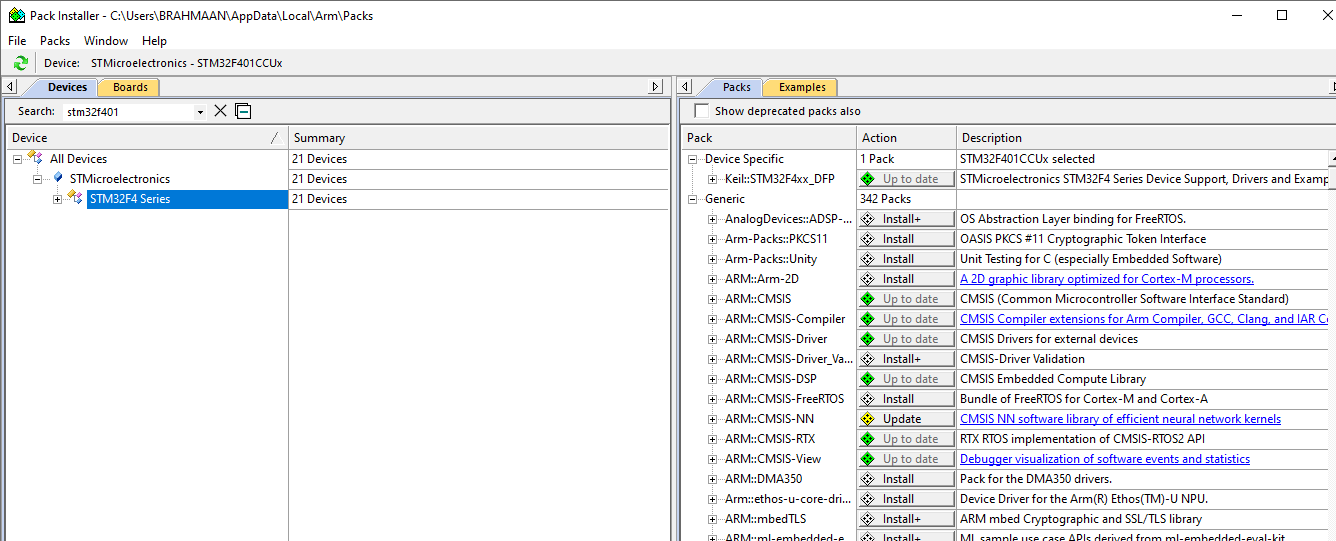
Note the progress information at the bottom.

1. After the selected items have downloaded, expand the NXP selection shown under the Devices tab in the left hand part of the window.

The expanded view might look like this:

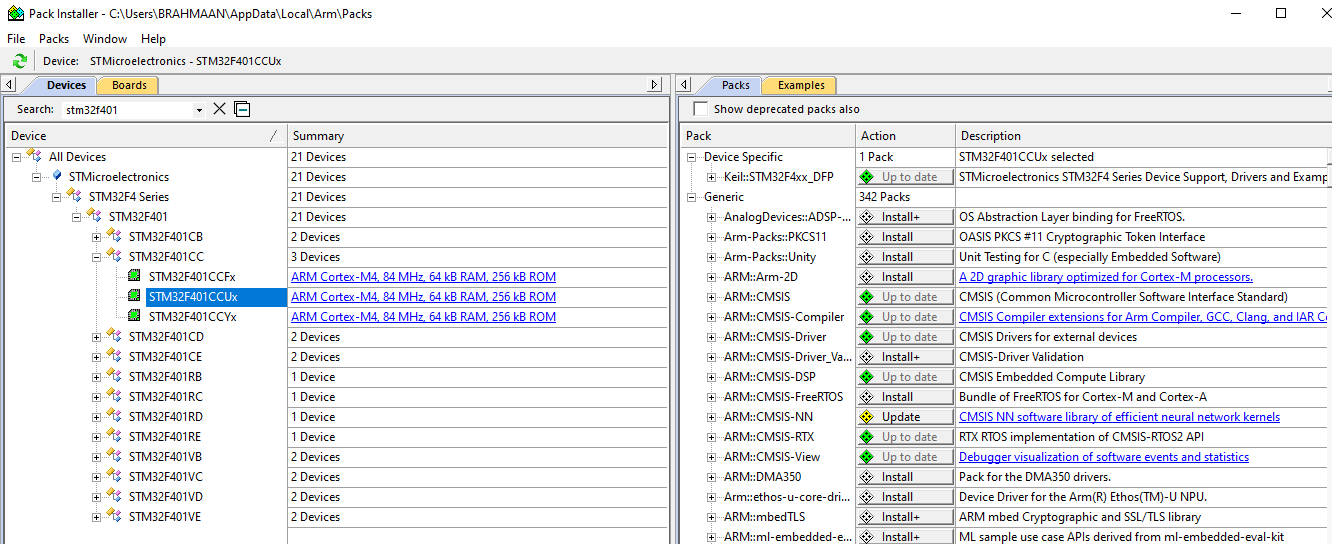
Click to expand the STM32F4 Series selection.

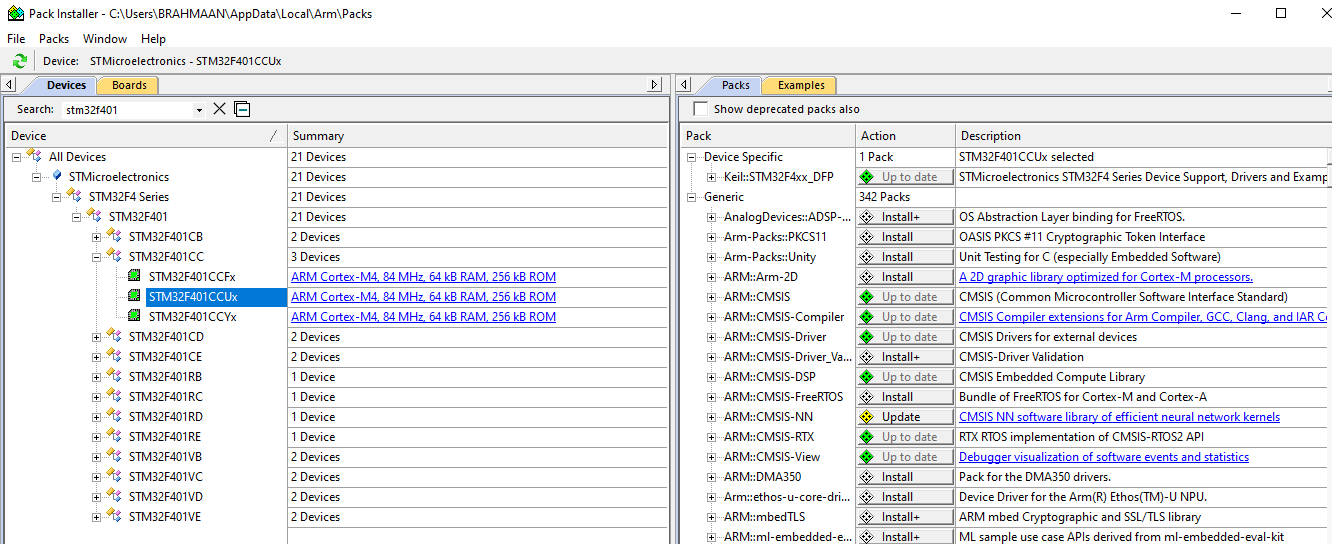
.



And then under the Packs tab click on the + box left of Device Specific.

Our interest is the STM32F401CCU6. Click on it in under the Devices tab



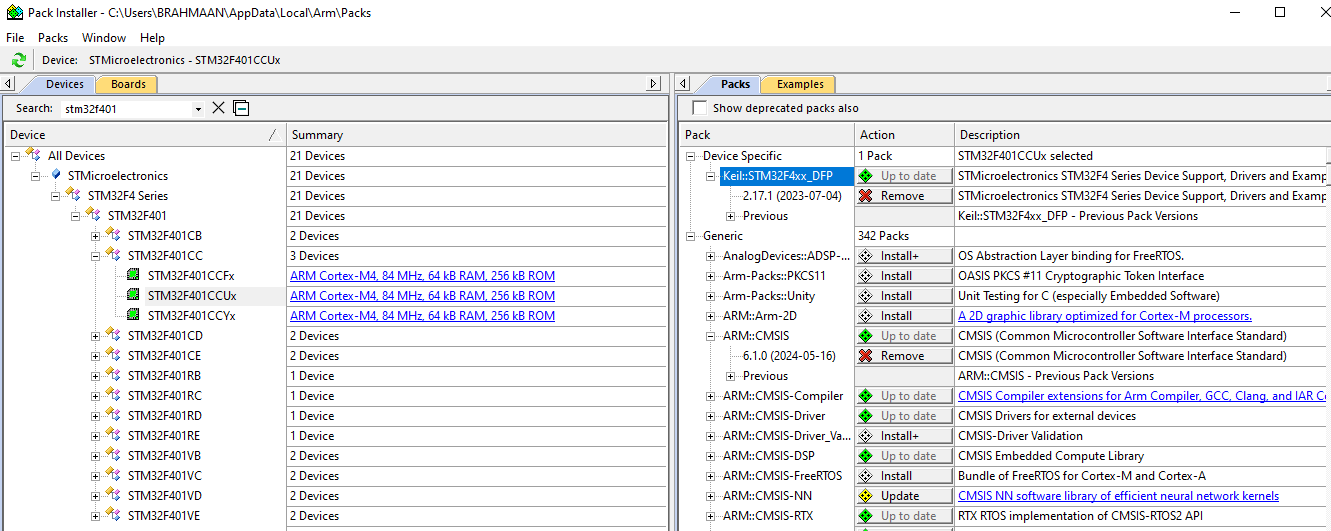
1. Click Install for STM32F4 Series Device Support and Examples:
2. 

The CMSIS and ARM Compiler packs have green diamonds by them because they are up to date.

After the Device Support and Examples pack is loaded the display will change as shown on the next page.

Note that a Remove option now is displayed for the Device Support and Examples package. That is normal and ok.

(in this screen snapshot the Bundle of FreeRTOS for Cortex-M and Cortex-A is not installed, but if that is displayed at this point go ahead and install it.)



That completes the installation of software and libraries that are needed. Close the installer.

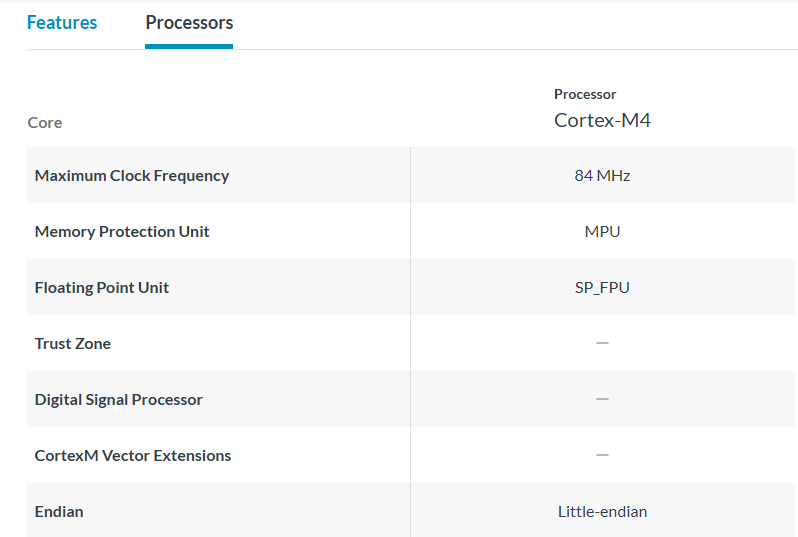
The next step is to create a project and work on a design. See a companion document for info regarding project creation.

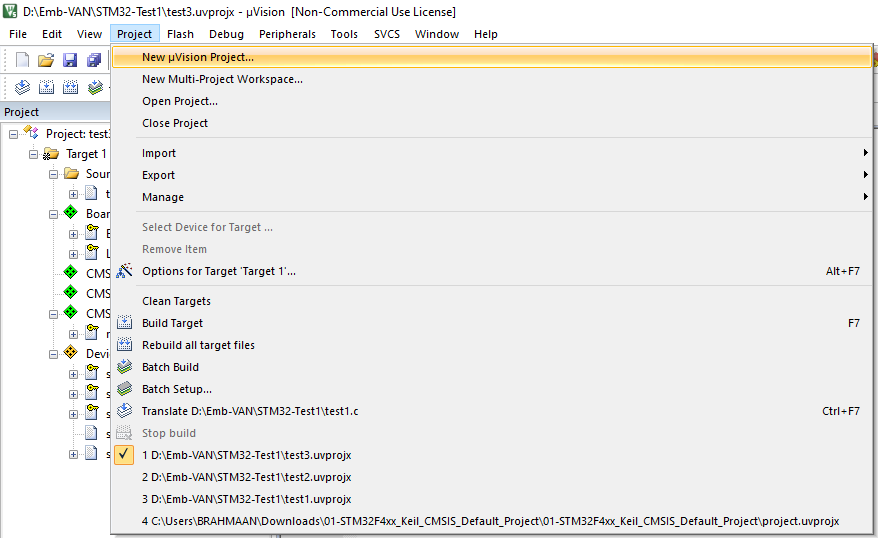
**STM32F401CCUx** [STMicroelectronics](https://www.keil.arm.com/vendors/stmicroelectronics/)

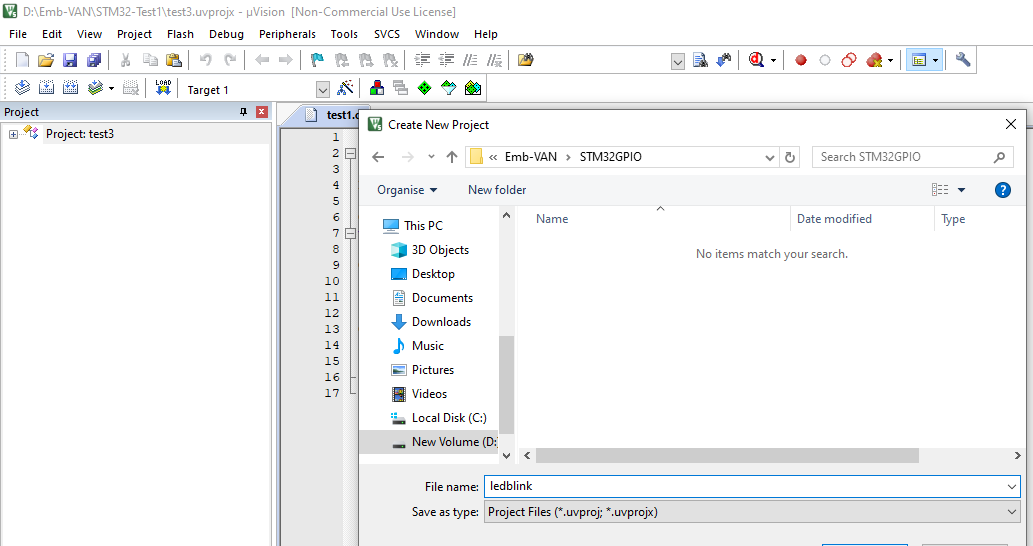
* **Core** Cortex-M4, 84 MHz
* **Family** STM32F4 Series
* **Sub-Family** STM32F401
* **CMSIS Pack** [STM32F4xx\_DFP](https://www.keil.arm.com/packs/stm32f4xx_dfp-keil/)

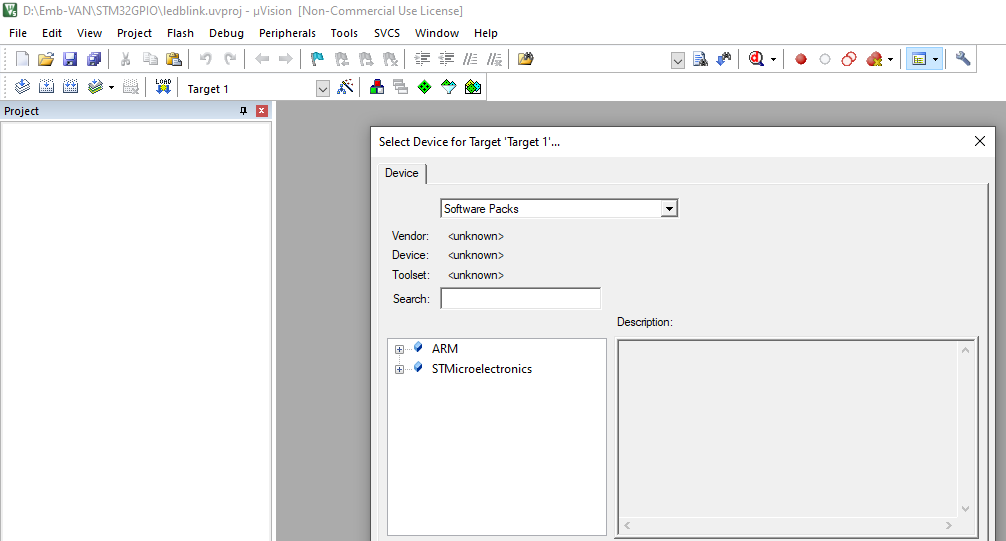
The STM32F4 family incorporates high-speed embedded memories and an extensive range of enhanced I/Os and peripherals connected to two APB buses, three AHB buses and a 32-bit multi-AHB bus matrix.  
  
- 64-Kbyte of CCM (core coupled memory) data RAM  
- LCD parallel interface, 8080/6800 modes  
- Timer with quadrature (incremental) encoder input  
- 5 V-tolerant I/Os  
- Parallel camera interface  
- True random number generator  
- RTC: subsecond accuracy, hardware calendar  
- 96-bit unique ID

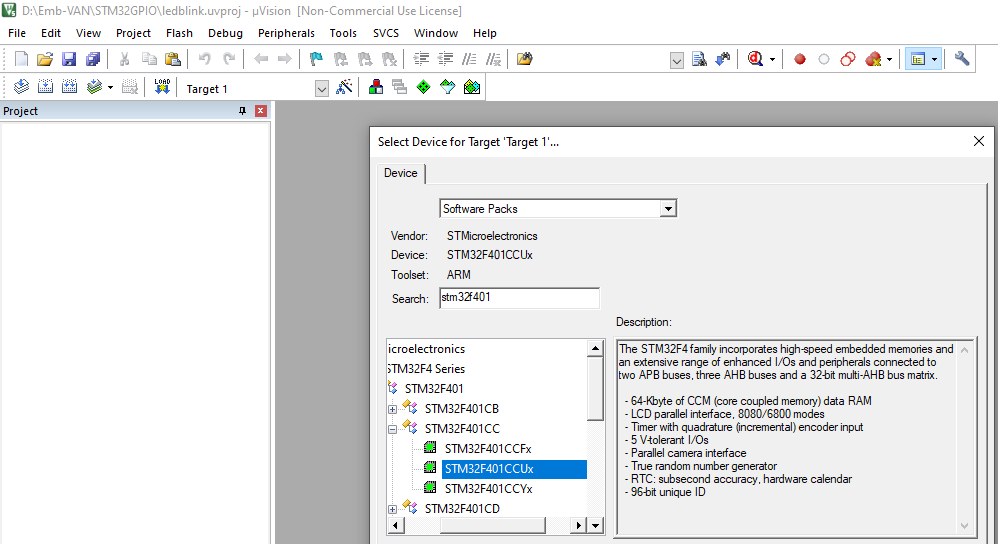
| **Memory** | |
| --- | --- |
| **IRAM1** | 64 KiB |
| **IROM1** | 256 KiB |

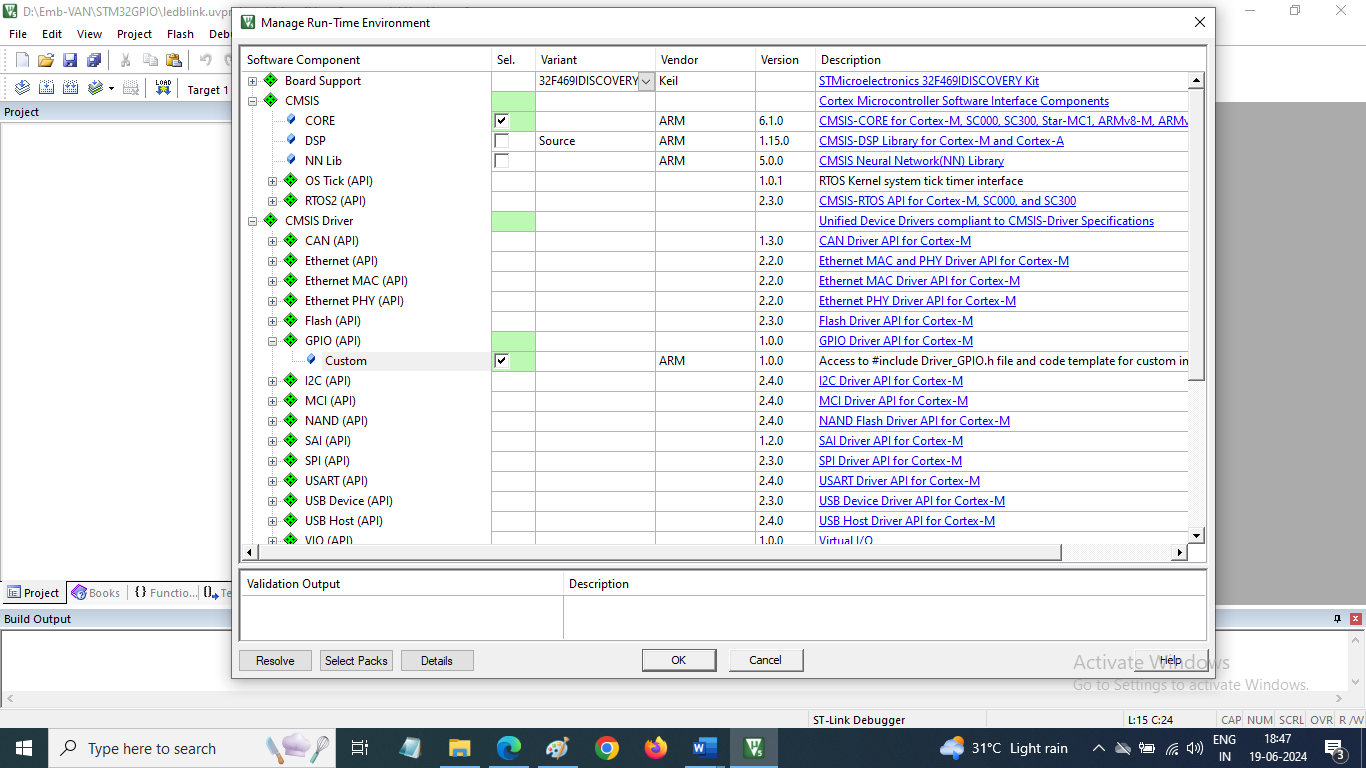


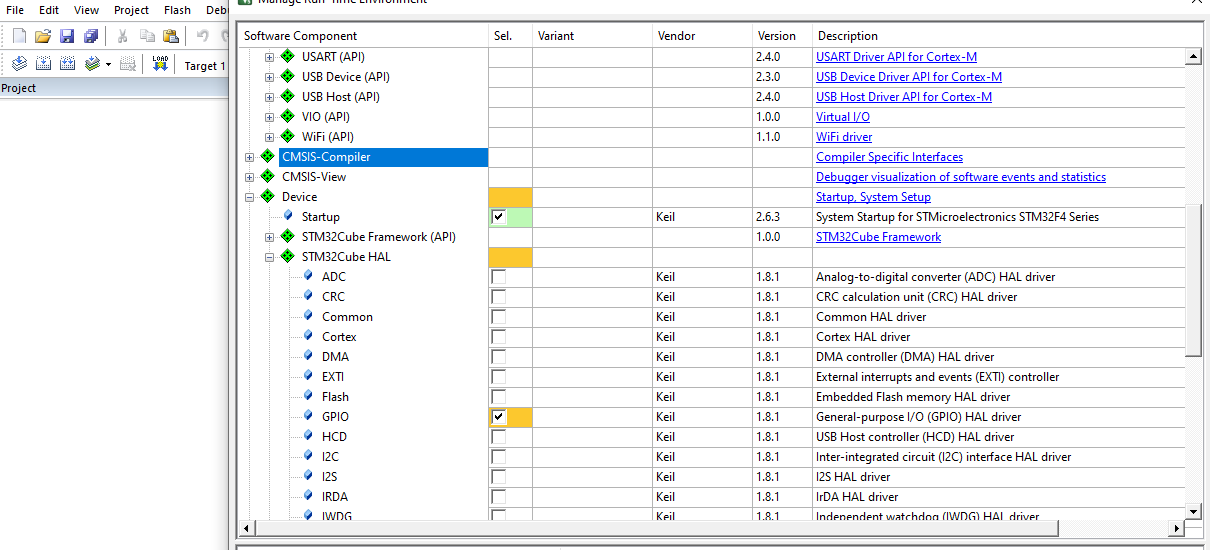


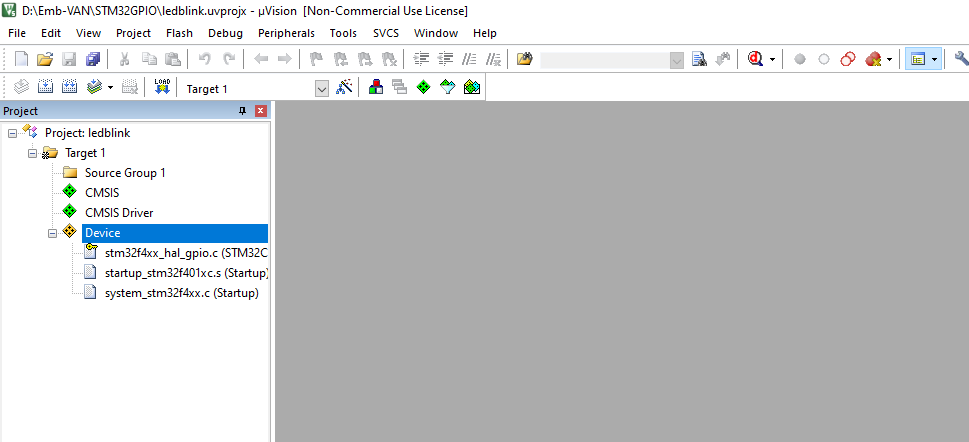


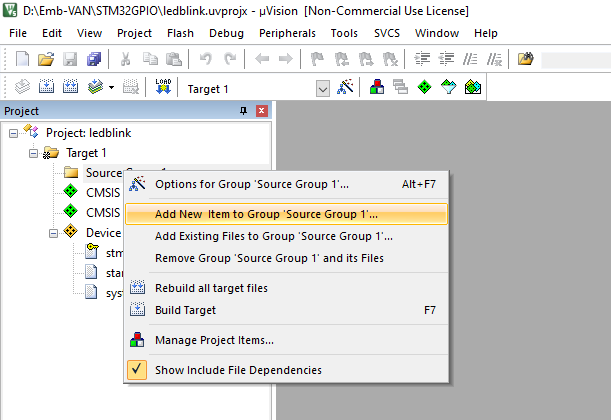


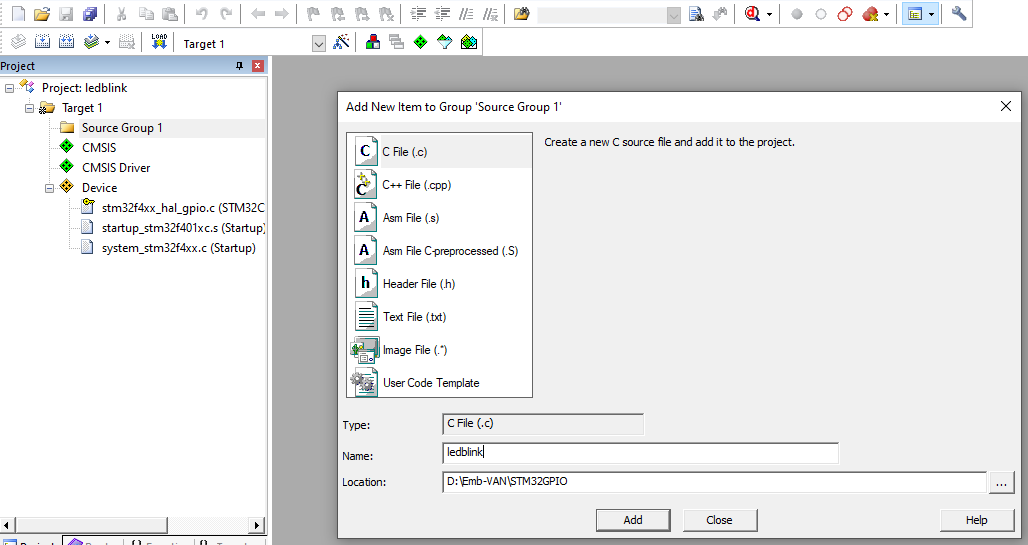


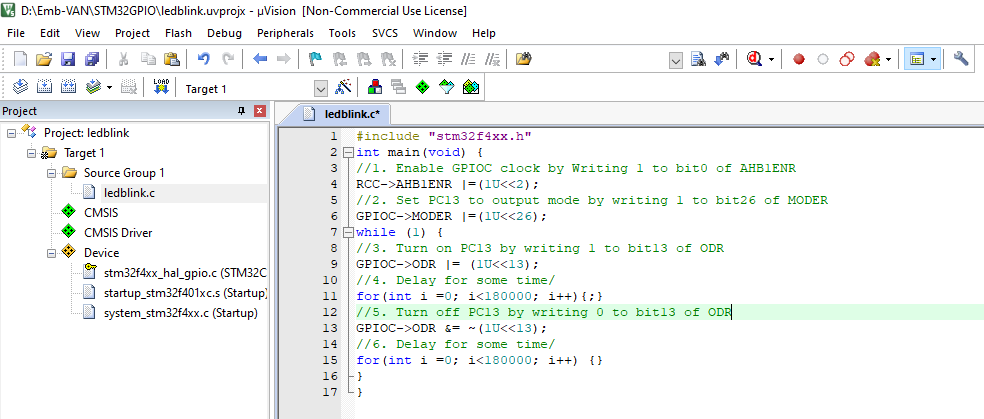


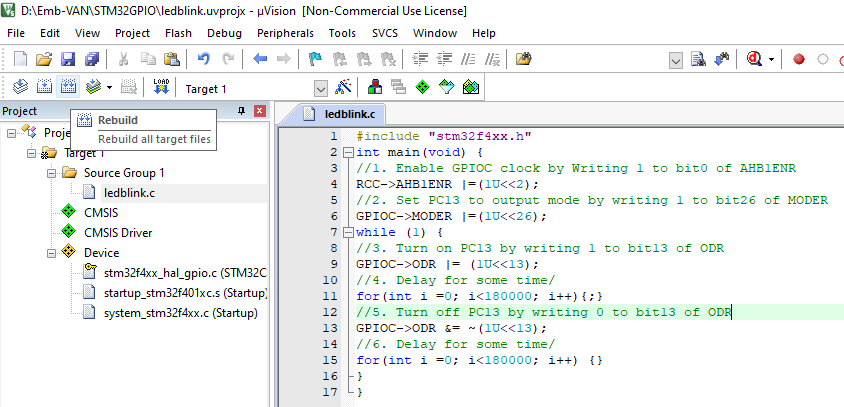


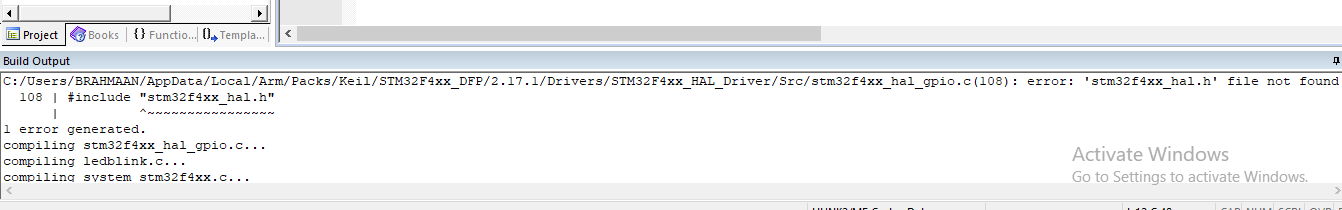


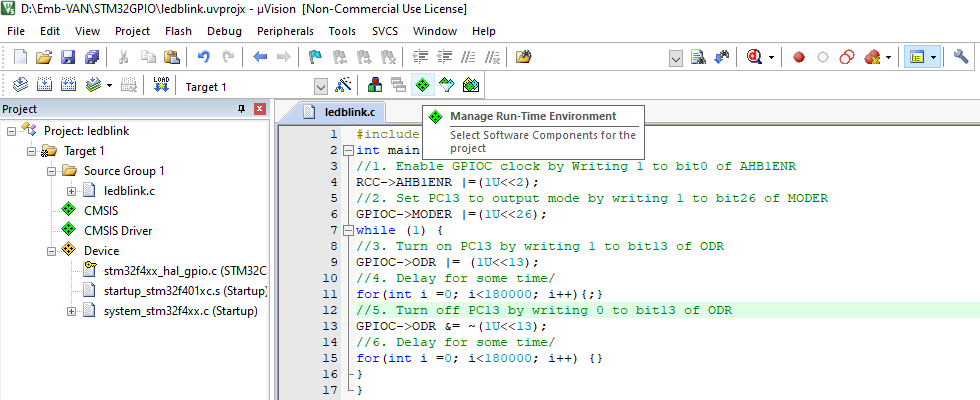


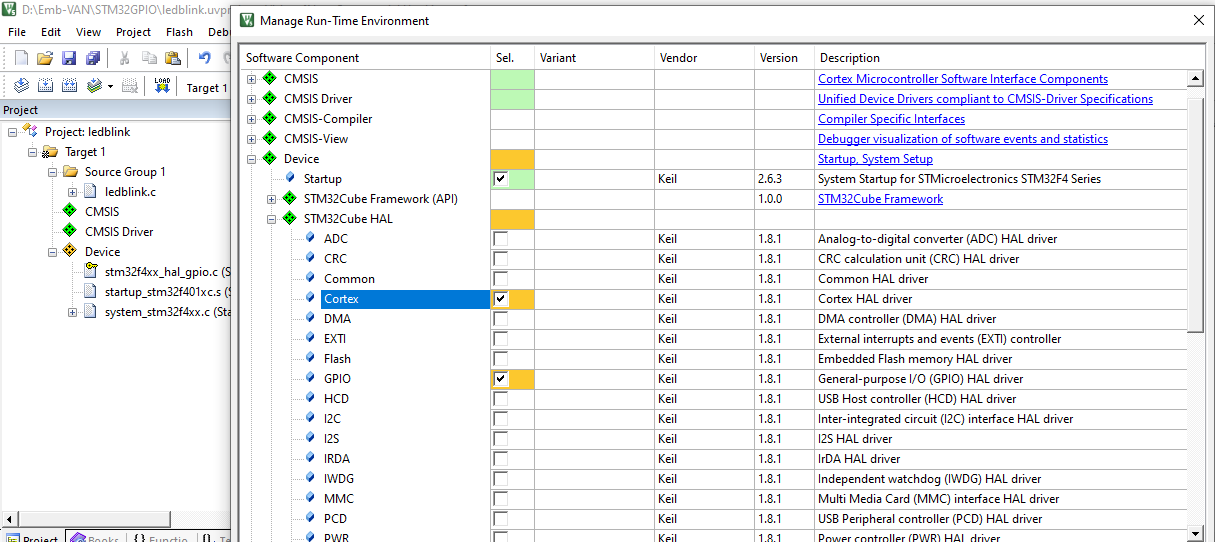


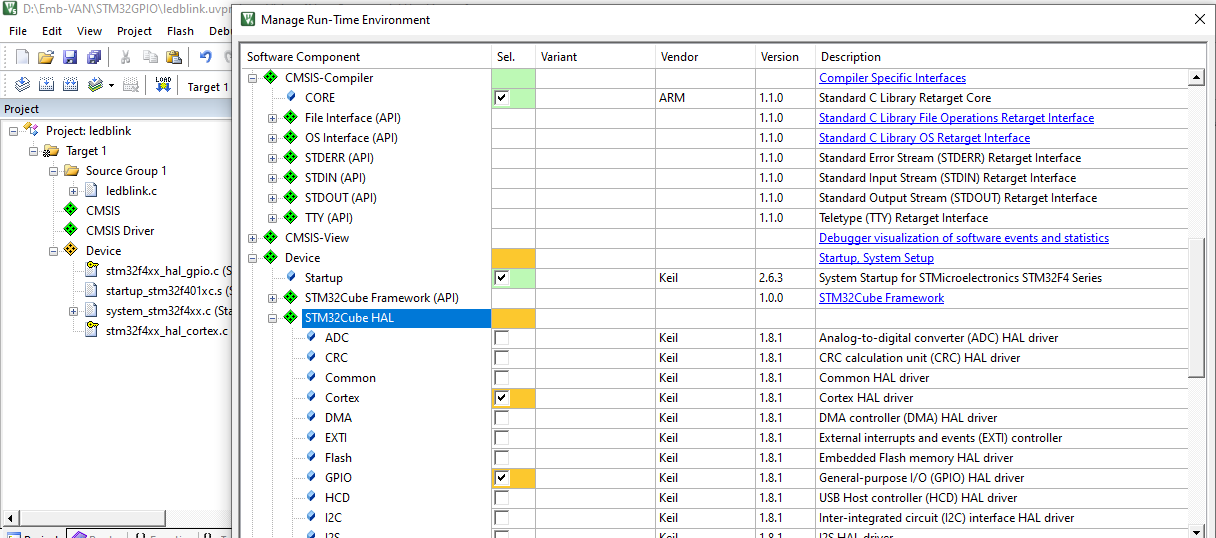


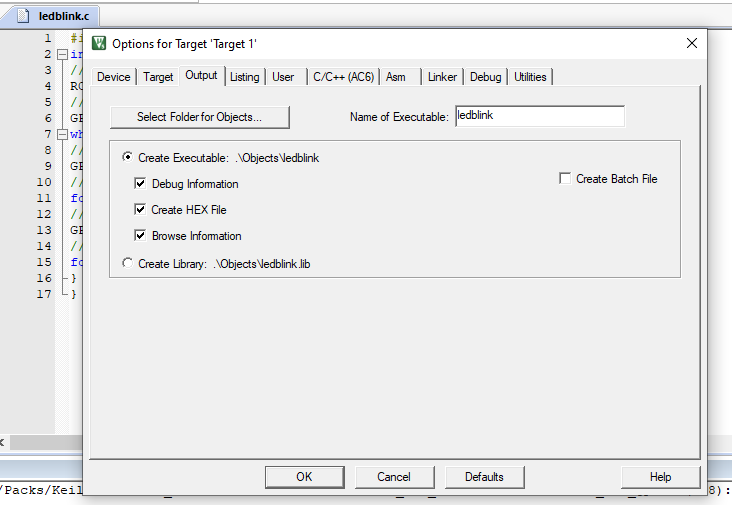


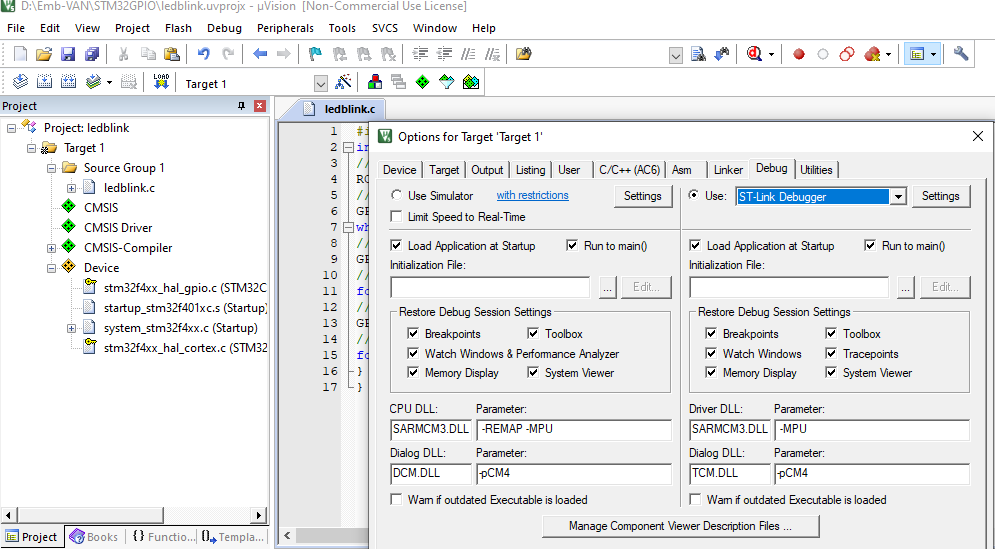


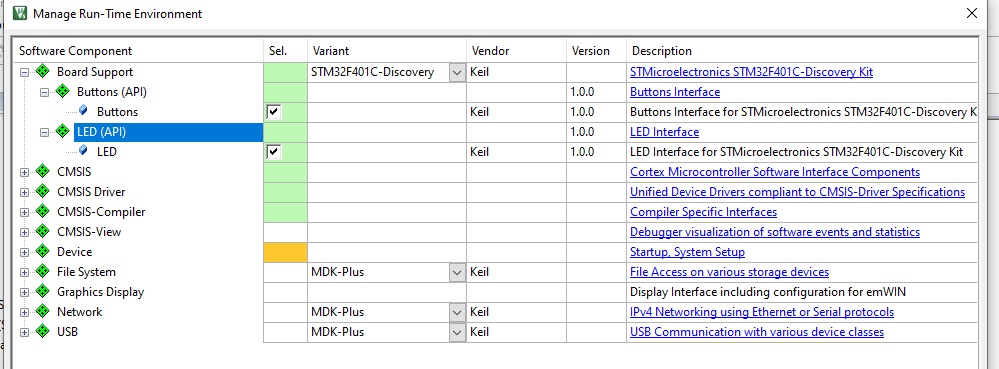


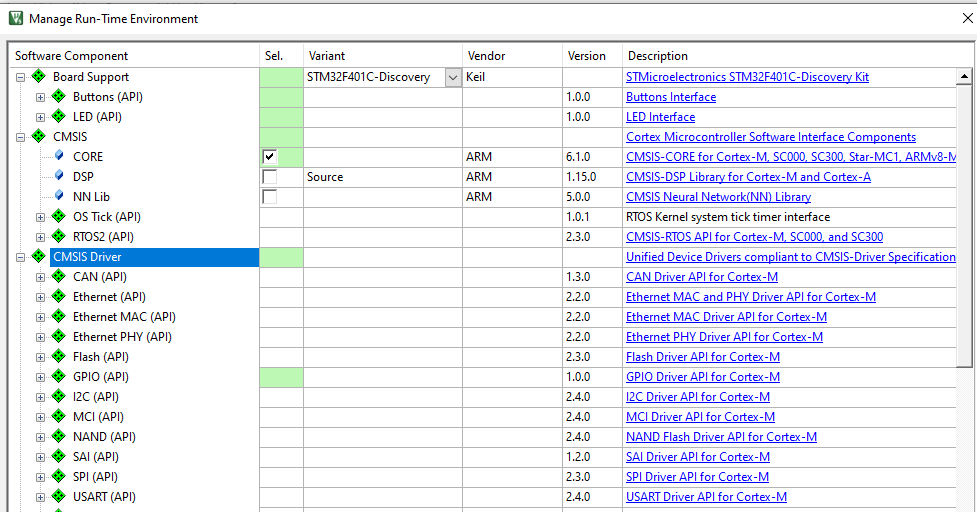


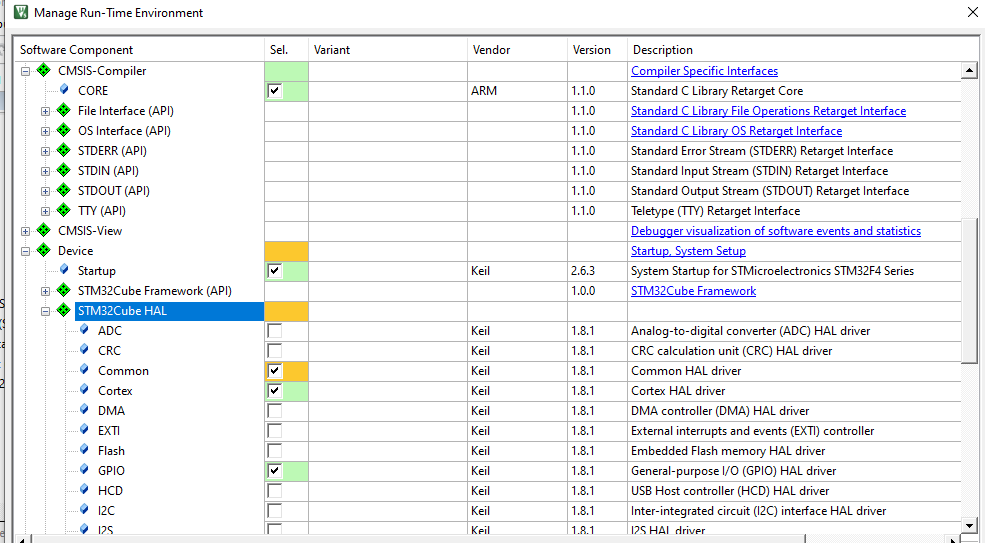


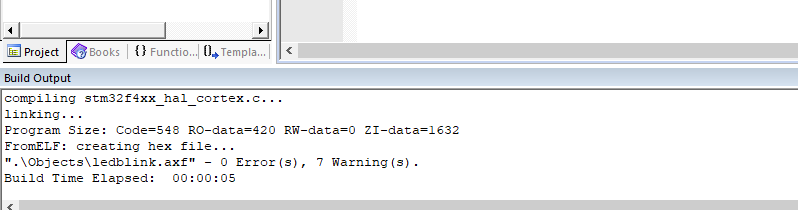


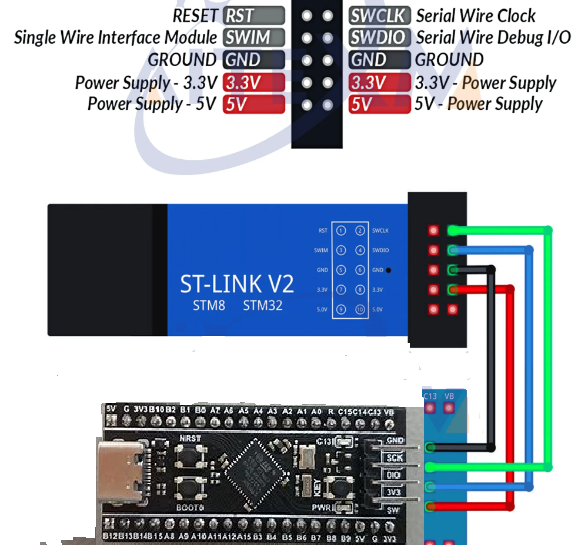


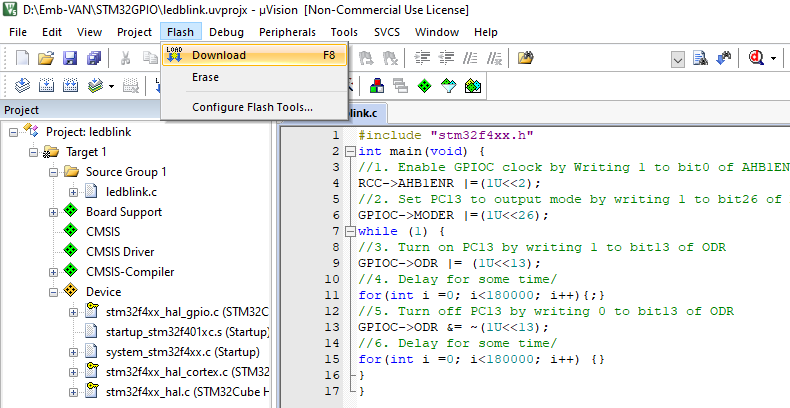


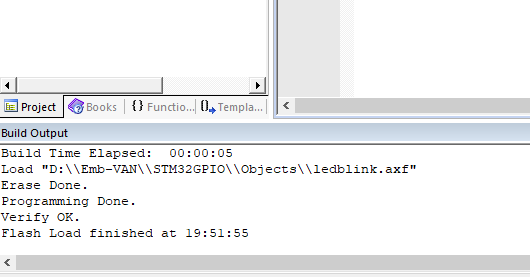


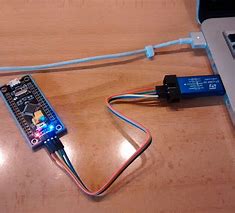












#include "stm32f4xx.h"

int main(void) {

//1. Enable GPIOC clock by Writing 1 to bit0 of AHB1ENR

RCC->AHB1ENR |=(1U<<2);

//2. Set PC13 to output mode by writing 1 to bit26 of MODER

GPIOC->MODER |=(1U<<26);

while (1) {

//3. Turn on PC13 by writing 1 to bit13 of ODR

GPIOC->ODR |= (1U<<13);

//4. Delay for some time/

for(int i =0; i<180000; i++){;}

//5. Turn off PC13 by writing 0 to bit13 of ODR

GPIOC->ODR &= ~(1U<<13);

//6. Delay for some time/

for(int i =0; i<180000; i++) {}

}

}

Keil uVision 5 and STM32F401CCU6 Mini Development Board

Blinking on board LED at PC13

#include "stm32f4xx.h"

int main(void) {

/\*1. Enable GPIOA clock by Writing 1 to bit0 of AHB1ENR\*/

RCC->AHB1ENR |=(1U<<2);

/\*2. Set PA5 to output mode by writing 1 to bit10 of MODER\*/

GPIOC->MODER |=(1U<<26);

while (1) {

/\*3. Turn on PA5 by writing 1 to bit5 of ODR\*/

GPIOC->ODR |= (1U<<13);

/\*4. Delay for some time\*/

for(int i =0; i<180000; i++){;}

/\*5. Turn off PA5 by writing 0 to bit5 of ODR\*/

GPIOC->ODR &= ~(1U<<13);

/\*6. Delay for some time\*/

for(int i =0; i<180000; i++) {}

}

}

A screenshot of a computer

Description automatically generated