

---

Home / General Purpose Microcontrollers

/ < Kinetis Microcontrollers Knowledge Base / OpenSDAv2

# OpenSDAv2

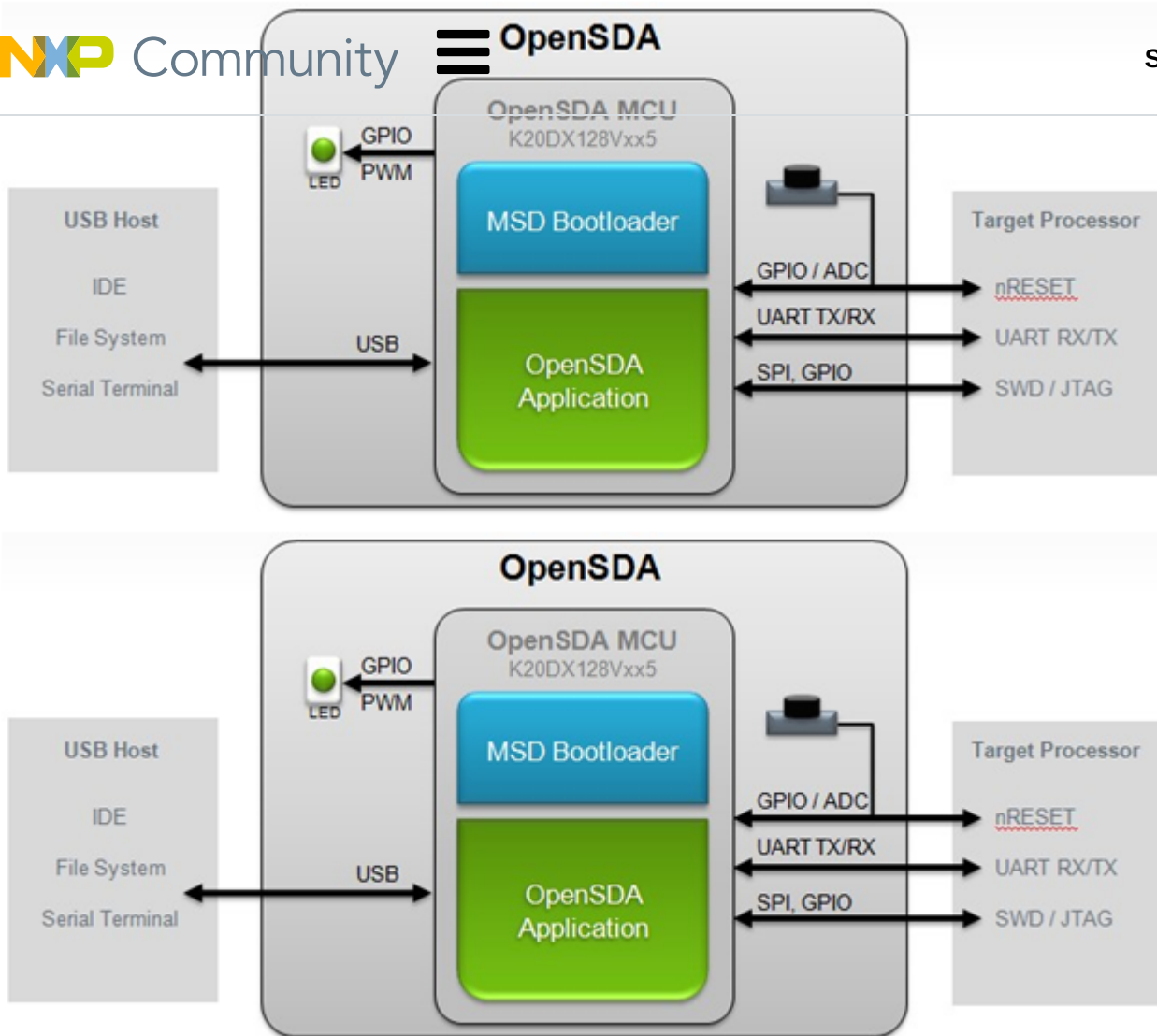
Search all content

Options

No ratings

OpenSDA/OpenSDAv2 is a serial and debug adapter that is built into several Freescale evaluation boards. It provides a bridge between your computer (or other USB host) and the embedded target processor, which can be used for debugging, flash programming, and serial communication, all over a simple USB cable.

The OpenSDA hardware consists of a circuit featuring a Freescale Kinetis K20 microcontroller (MCU) with an integrated USB controller. On the software side, it implements a mass storage device bootloader which offers a quick and easy way to load OpenSDA applications such as flash programmers, run-control debug interfaces, serial to USB converters, and more. Details on OpenSDA can be found in the [OpenSDA User Guide](#).



The bootloader and app firmware that lay on top of the original OpenSDA circuit was proprietary. But recently ARM decided to open source their CMSIS-DAP interface, and now a truly open debug platform could be created. This new open-sourced firmware solution is known as OpenSDAv2.

## OpenSDAv2:

OpenSDAv2 uses the exact same hardware circuit as the original OpenSDA solution, and out of the box it still provides a debugger, drag-and-drop flash programmer, and virtual serial port over a single USB cable.

The difference is the firmware implementation:

- **OpenSDA:** Programmed with the proprietary P&E Micro developed bootloader. P&E Micro is the default debug interface app.
- **OpenSDAv2:** Programmed with the open-sourced CMSIS-DAP/mbed bootloader. CMSIS-DAP is the default debug interface app.

	Firmware Developer	Kinetis K20 Based Hardware Circuit	Default Debug Interface	Drag-and-drop Target MCU Flash Programming	Virtual Serial Port	Source Code Available
OpenSDA	P&E Micro	x	P&E Micro	.srec/.s19	x	
OpenSDAv2	ARM/mbed.org	x	CMSIS-DAP	.bin	x	x

The bootloader and app firmware used by OpenSDAv2 is developed by the community at mbed.org, and is known as “[CMSIS-DAP Interface Firmware](#)”. If you explore that site, you will find that this firmware was also ported to run on other hardware, but the combination of this mbed.org firmware with the Kinetis K20 MCU is known as OpenSDAv2.

It is important to understand however that it is possible to run a P&E Micro debug app on the CMSIS-DAP/mbed bootloader found on OpenSDAv2. Likewise it is possible to run a CMSIS-DAP debug app on the P&E Micro bootloader found on OpenSDA. The debug application used needs to be targeted towards a specific bootloader though, as a single binary cannot be used on both the OpenSDA and OpenSDAv2 bootloaders.

## OpenSDAv2.1:

During development of OpenSDAv2 features and bug fixes, it was found that the reserved bootloader space was too small. Thus a new version of OpenSDAv2 had to be created, which was named OpenSDAv2.1. The difference between the OpenSDAv2.0 and v2.1 is the address where the debug application starts: for OpenSDAv2.0 it expects the application at address 0x5000, while OpenSDAv2.1 expects the application to start at address 0x8000.

The only board with OpenSDAv2.0 is the FRDM-K64F. All other OpenSDAv2 boards (such as the just released FRDM-K22F) use OpenSDAv2.1.

Unfortunately this means that new OpenSDAv2 apps are needed. From a user perspective this mostly affects the JLink app since it was shared across all boards. Make sure you download the correct app for your board based on the OpenSDAv2 version.

## OpenSDAv2 Apps:

- [mbed CMSIS-DAP for FRDM-K64F](#)

- [Segger JLink](#) (look at bottom of page for OpenSDAv2.0 or OpenSDAv2.1 app)

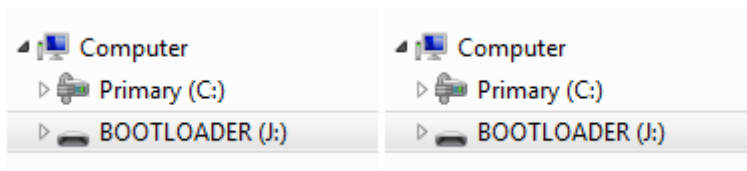
## OpenSDAv2 Bootloader:

The key difference between OpenSDA and OpenSDAv2 is the bootloader.





- Boards with **OpenSDA** use a proprietary bootloader developed by P&E Micro, and it cannot be erased or reprogrammed by an external debugger due to the security restrictions in the firmware.
- Boards with **OpenSDAv2** use the open-source bootloader developed by mbed.org, and it can be erased and reprogrammed with an external debugger.



Apps need to be specifically created to work with either the P&E bootloader (Original OpenSDA) or the CMSIS-DAP/mbed bootloader (OpenSDAv2/OpenSDAv2.1) as the bootloader memory map is different. Thus it's important to know which type of bootloader is on your board to determine which version of an app to load.

You can determine the bootloader version by holding the reset button while plugging in a USB cable into the OpenSDA USB port. A **BOOTLOADER** drive will appear for both OpenSDA and OpenSDAv2.



The OpenSDAv2.0 bootloader (may also be called the CMSIS-DAP/mbed bootloader) developed by mbed.org will have the following files inside. Viewing the HTML source of the bootload.htm file with Notepad will tell you the build version, date, and git hash commit. For the OpenSDAv2.1 bootloader, this file will be named mbed.htm instead.

Name	Date modified	Type	Size
 .fseventsd	12/14/2012 1:43 PM	File folder	
 .metadata_never_index	12/14/2012 1:44 PM	METADATA_NEVE...	0 KB
 .Trashes	12/14/2012 1:44 PM	TRASHES File	0 KB
 bootload.htm	12/14/2012 1:52 PM	Firefox HTML Doc...	1 KB

SIGN IN

Name	Date modified	Type	Size
.fsevents	12/14/2012 1:43 PM	File folder	
.metadata_never_index	12/14/2012 1:44 PM	METADATA_NEVE...	0 KB
.Trashes	12/14/2012 1:44 PM	TRASHES File	0 KB
bootload.htm	12/14/2012 1:52 PM	Firefox HTML Doc...	1 KB

The OpenSDAv1 bootloader developed by P&E Micro will have the following inside. Clicking on SDA\_INFO.HTM will take you to the P&E website.

Name	Date modified	Type	Size
FSL_WEB.HTM	8/8/2012 9:58 PM	Firefox HTML Doc...	1 KB
LASTSTAT.TXT	8/8/2012 9:58 PM	TXT File	1 KB
SDA_INFO.HTM	8/8/2012 9:58 PM	Firefox HTML Doc...	2 KB
TOOLS.HTM	8/8/2012 9:58 PM	Firefox HTML Doc...	1 KB

Name	Date modified	Type	Size
FSL_WEB.HTM	8/8/2012 9:58 PM	Firefox HTML Doc...	1 KB
LASTSTAT.TXT	8/8/2012 9:58 PM	TXT File	1 KB
SDA_INFO.HTM	8/8/2012 9:58 PM	Firefox HTML Doc...	2 KB
TOOLS.HTM	8/8/2012 9:58 PM	Firefox HTML Doc...	1 KB

## Using CMSIS-DAP:

When you connect a Freedom board that has OpenSDAv2 (such as the FRDM-K64F) to your computer with a USB cable, it will begin running the default CMSIS\_DAP/mbed application which has three main features.

### 1. Drag and Drop MSD Flash Programming

You will see a new disk drive appear labeled “MBED”.



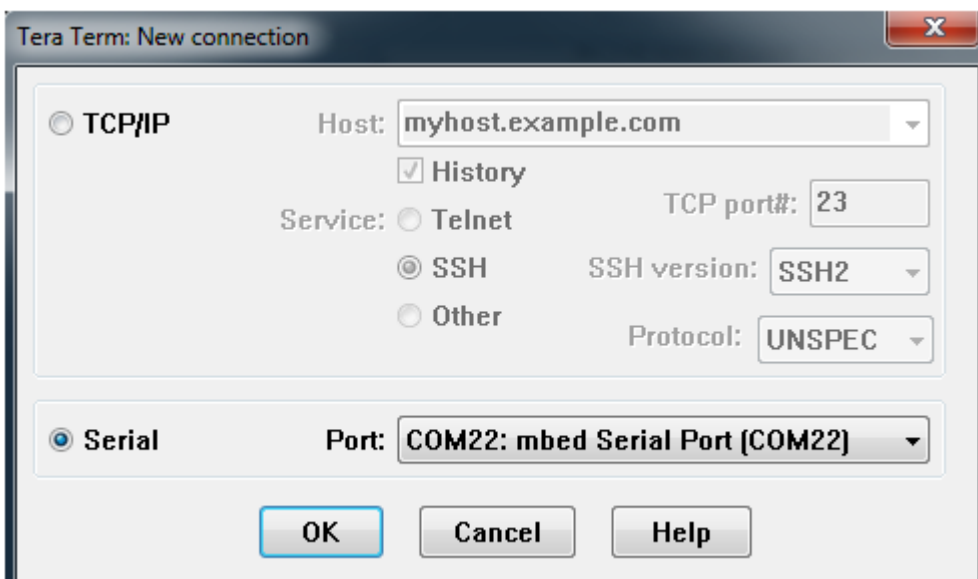
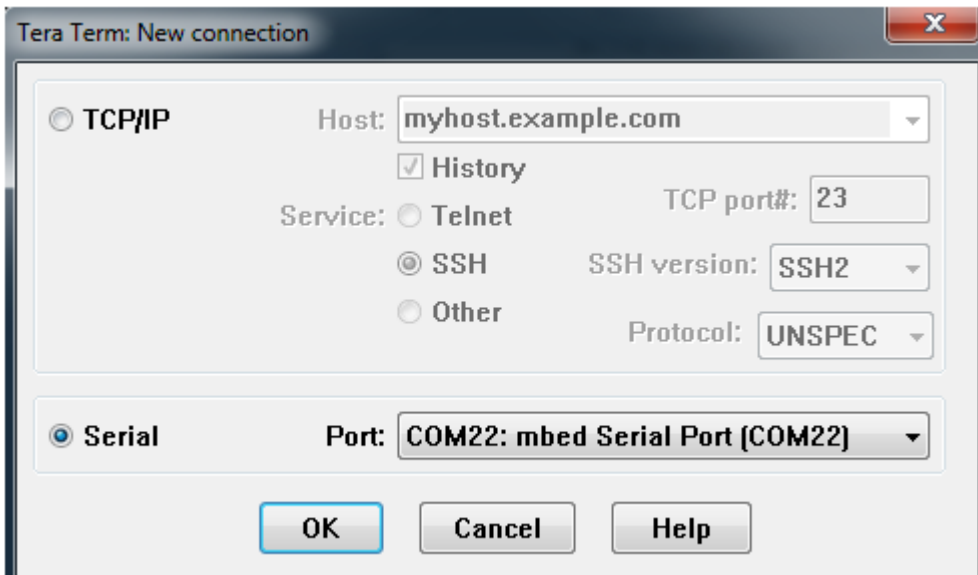
You can then drag-and-drop binary (.bin) files onto the virtual hard disk to program the target MCU.



[SIGN IN](#)

## 2.Virtual Serial Port

OpenSDAv2 will also enumerate as a virtual serial port, which you can use a terminal program, such as TeraTerm (shown below), to connect to. You may need to install the [mbed Windows serial port driver](#) first before the serial port will enumerate on Windows properly. It should work without a driver for MacOS and Linux.



## 3. Debugging

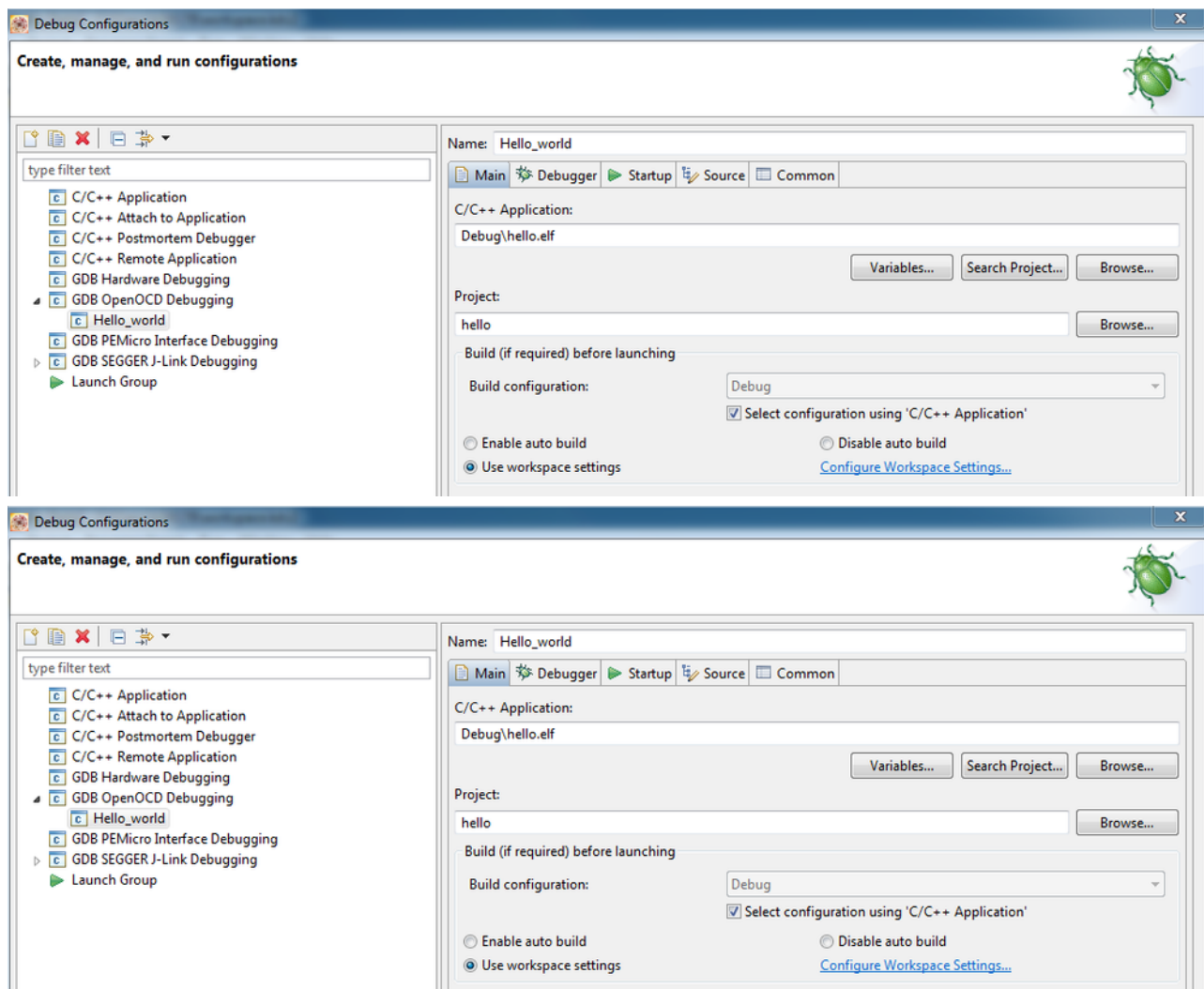
The CMSIS-DAP app also allows you to debug the target MCU via the CMSIS-DAP interface. Select the CMSIS-DAP interface in your IDE of choice, and inside the CMSIS-DAP options select the Single Wire Debug (SWD) option:

**Kinetis Design Studio (KDS):**

Note: OpenOCD with CMSIS-DAP for FRDM-K22F is **not** supported in KDS V1.1.0. You must use either the P&E app instructions or the JLink app instructions to use KDS with the FRDM-K22F at this time. This will be fixed over the next few weeks.

OpenSDAv2 uses the OpenOCD debug interface which uses the CMSIS-DAP protocol.

Make sure '-f kinetis.cfg' is specified as 'Other Options':



☒ Start OpenOCD locally

Executable:

Browse...

Variables...

GDB port:

Telnet port:

Log file:

Browse...

Other options:

☒ Allocate console for OpenOCD

☐ Allocate console for the telnet connection

#### GDB Client Setup

Executable:

Browse...

Variables...

Other options:

Commands:

#### Debug Options

☐ Connect to running target

#### Remote Target

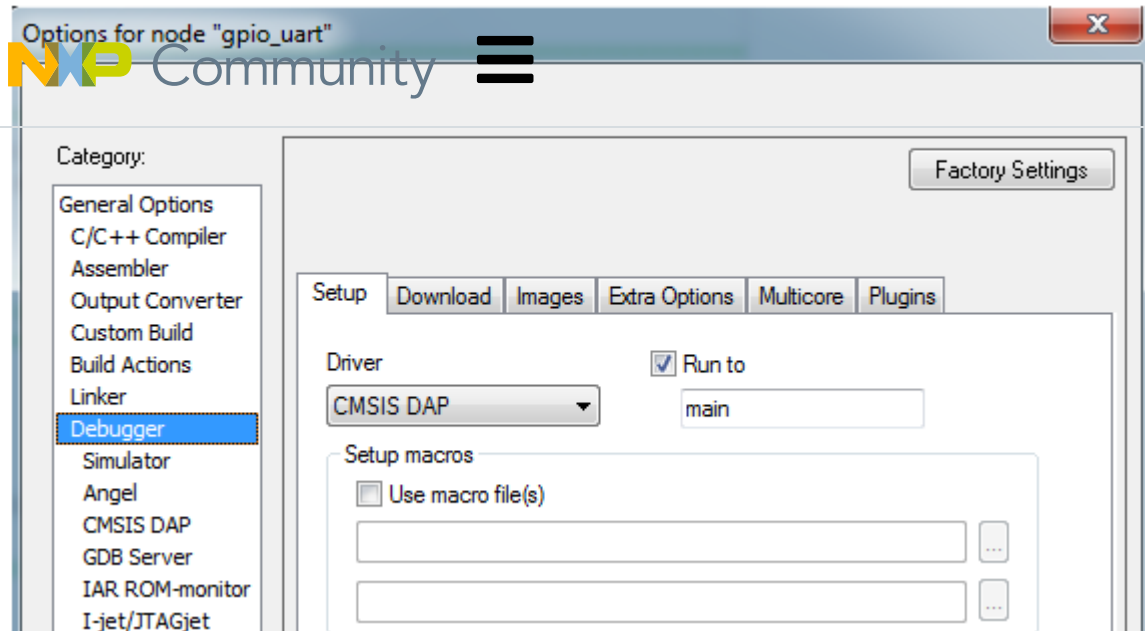
Host name or IP address:

Port number:

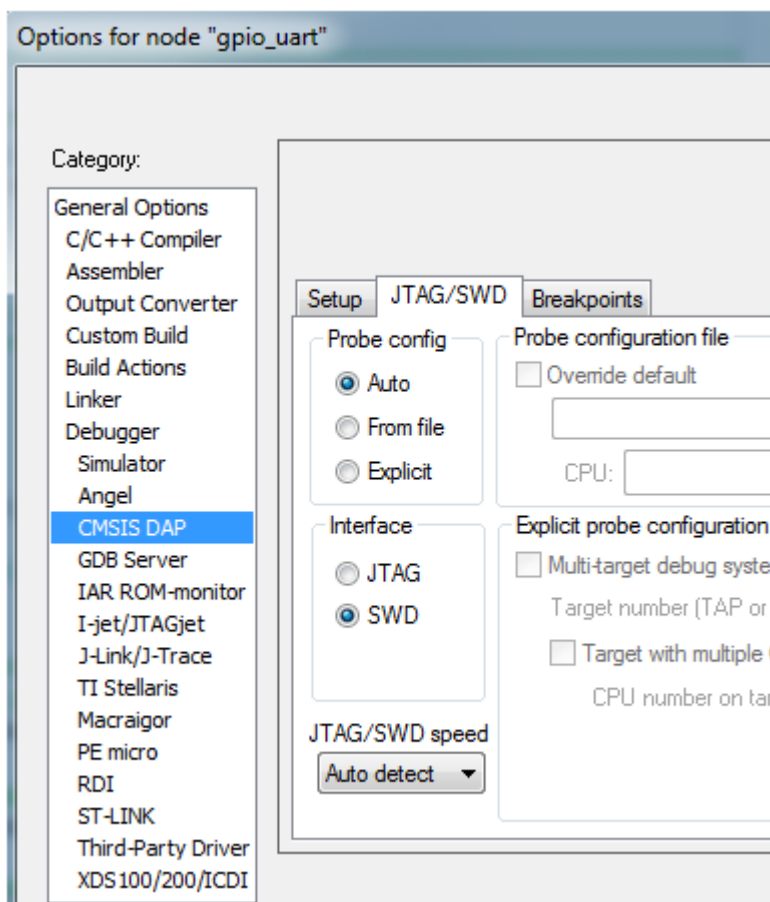
☐ Force thread list update on suspend

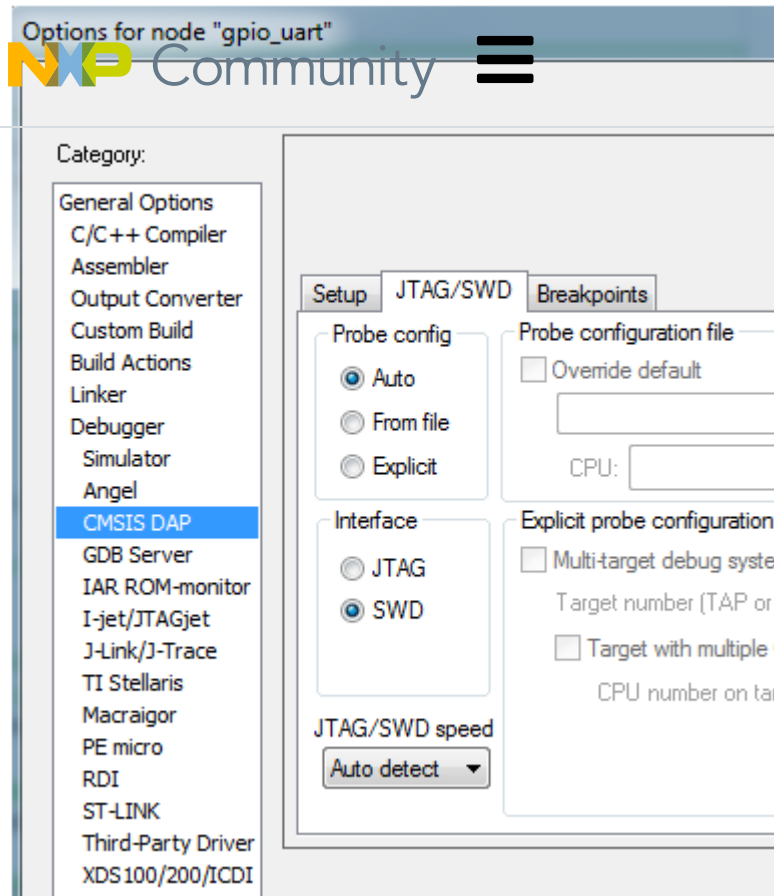






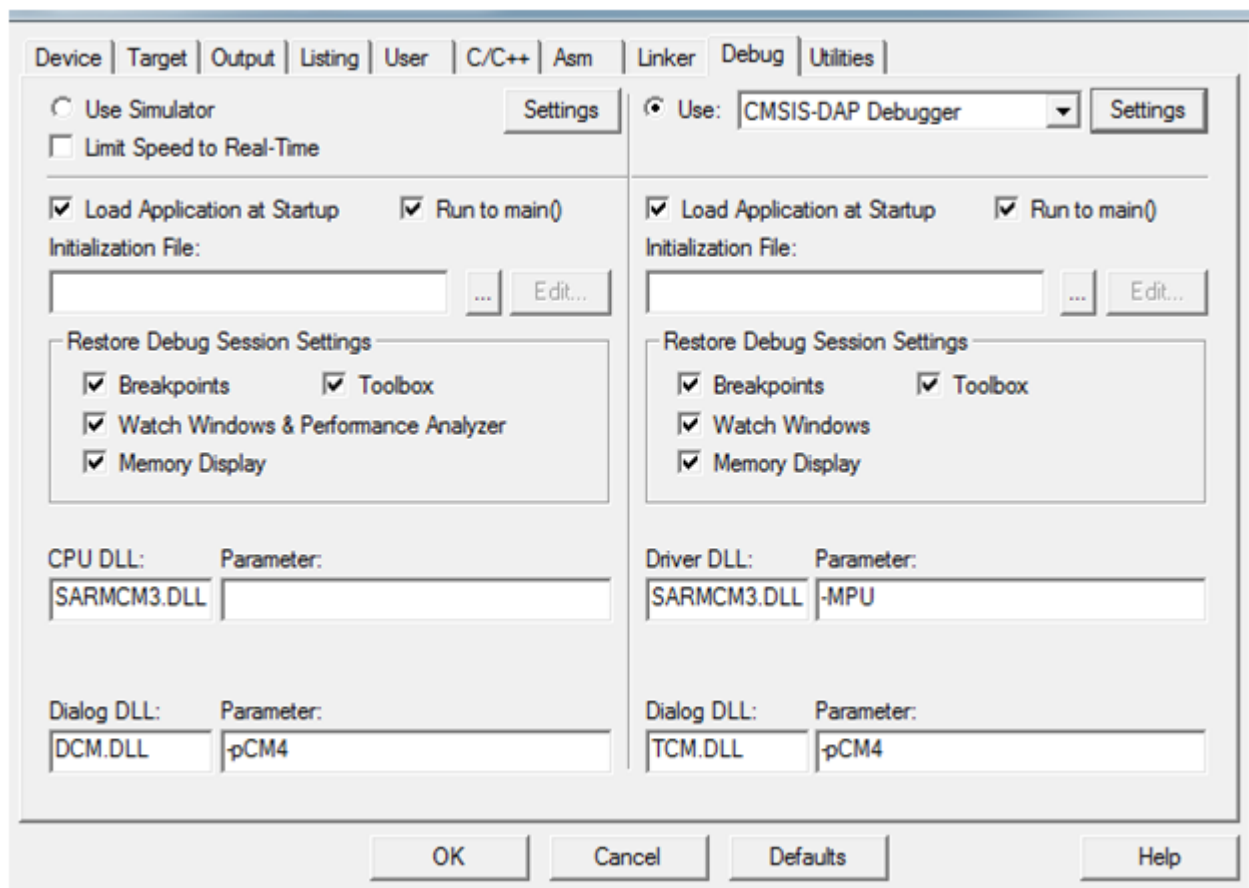
[SIGN IN](#)





[SIGN IN](#)

Keil:



The image shows the 'CMSIS-DAP Debugger' settings window in Keil MDK-ARM. The window is divided into two panes. The left pane has tabs for 'Target', 'Output', 'Listing', 'User', 'CMSIS-DAP', 'Asm', 'Linker', 'Debug', and 'Utilities'. The 'CMSIS-DAP' tab is selected. It contains options for 'Use Simulator' (unchecked), 'Limit Speed to Real-Time' (unchecked), 'Load Application at Startup' (checked), 'Run to main()' (checked), and an 'Initialization File' field. Below these is a 'Restore Debug Session Settings' section with checkboxes for 'Breakpoints', 'Toolbox', 'Watch Windows & Performance Analyzer', and 'Memory Display'. At the bottom are fields for 'CPU DLL' (SARMCM3.DLL) and 'Parameter' (empty).

The right pane shows the 'Use:' dropdown set to 'CMSIS-DAP Debugger'. It has similar options for 'Load Application at Startup' (checked), 'Run to main()' (checked), and 'Initialization File'. The 'Restore Debug Session Settings' section is also present. At the bottom are fields for 'Driver DLL' (SARMCM3.DLL) and 'Parameter' (-MPU). The 'Dialog DLL' is set to DCM.DLL and the 'Parameter' is -pCM4.

At the bottom of the window are buttons for 'OK', 'Cancel', 'Defaults', and 'Help'.

**Cortex-M Target Driver Setup**

Debug | **Flash Download**

**CMSIS-DAP - JTAG/SW Adapter**

MBED CMSIS-DAP

Serial No: 02400201B13F0E

Firmware Version: 1.0

☒ SWJ Port: SW

Max Clock: 1MHz

**SW Device**

IDCODE	Device Name	Move
<input checked="" type="radio"/> 0x2BA01477	ARM CoreSight SW-DP	Up Down

☒ Automatic Detection ID CODE:

☐ Manual Configuration Device Name:

Add Delete Update AP: 0x00

**Debug**

**Connect & Reset Options**

Connect: Normal Reset: Autodetect

☒ Reset after Connect

**Cache Options**

☒ Cache Code

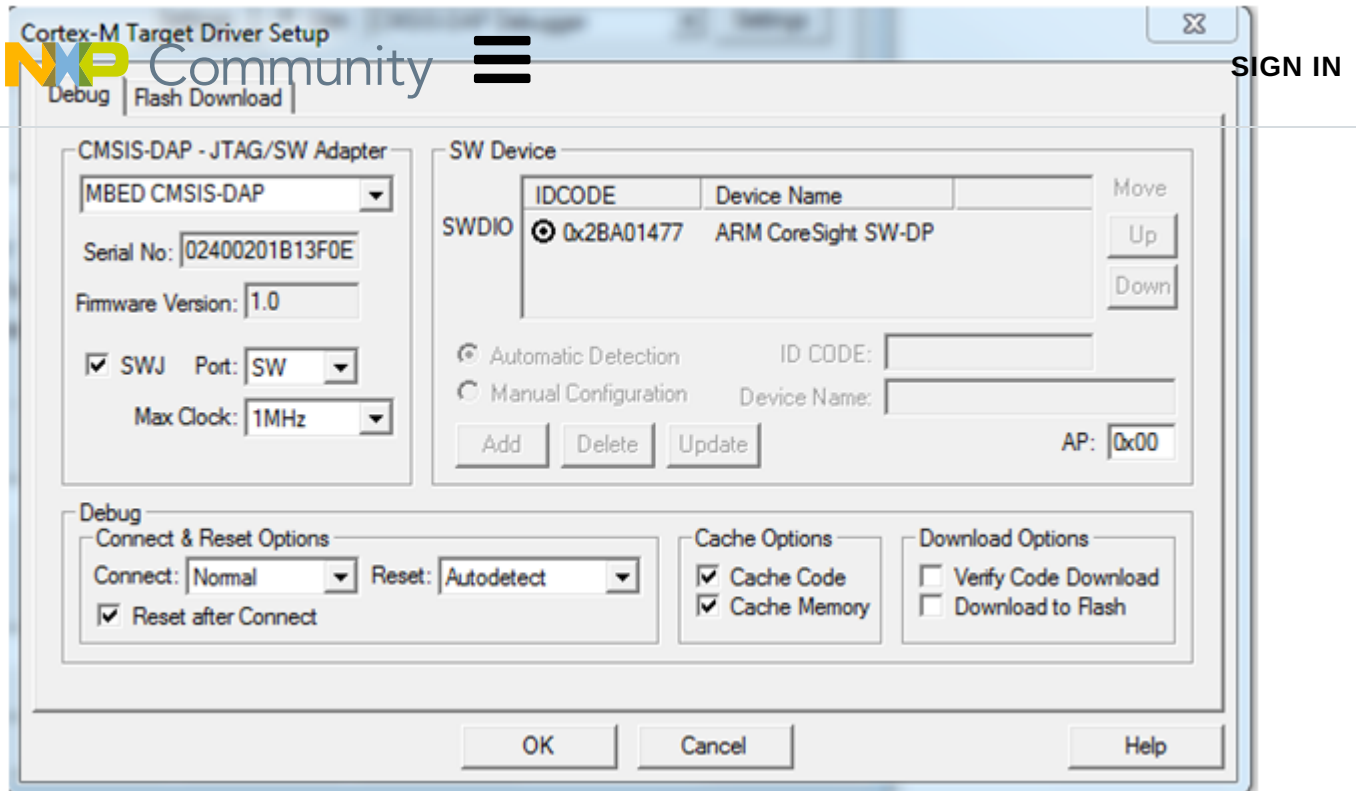
☒ Cache Memory

**Download Options**

☐ Verify Code Download

☐ Download to Flash

OK Cancel Help



## Resources

[CMSIS-DAP Interface Firmware](#)

[mbed.org FRDM-K64 Page](#)

[FRDM-K64 User Guide](#)

[OpenSDAv2 on MCU on Eclipse blog](#)

[OpenSDA User Guide](#)

[KDS Debugging](#)

## Appendix A: Building the CMSIS-DAP Debug Application

The open source CMSIS-DAP Interface Firmware app is the default app used on boards with OpenSDAv2. It provides:

- Debugging via the CMSIS-DAP interface
- Drag-and-drop flash programming
- Virtual Serial Port providing USB-to-Serial convertor

While binaries of this app are provided for supported boards, some developers would like to build the CMSIS-DAP debug application themselves.

This debug application can be built for either the OpenSDAv2/mbed bootloader, or for the

original OpenSDA bootloader developed by P&E Micro. If you are not sure which



to determine your board has, refer to the bootloader section in this document.

[SIGN IN](#)

Building the CMSIS-DAP debug application requires Keil MDK. You will also need to have the “Legacy Support for Cortex-M Devices” software pack installed for Keil.

You will also need Python 2.x installed. Due to the python script used, Python 3.x will not work.

The code is found in the MBED git repository, so it can be downloaded using a git clone command:

“**git clone** <https://github.com/mbedmicro/CMSIS-DAP.git>”

**Note that there is a Download Zip option, but you will run into a issue when trying to compile that version, so you must download it via git instead.**

The source code can be seen below:

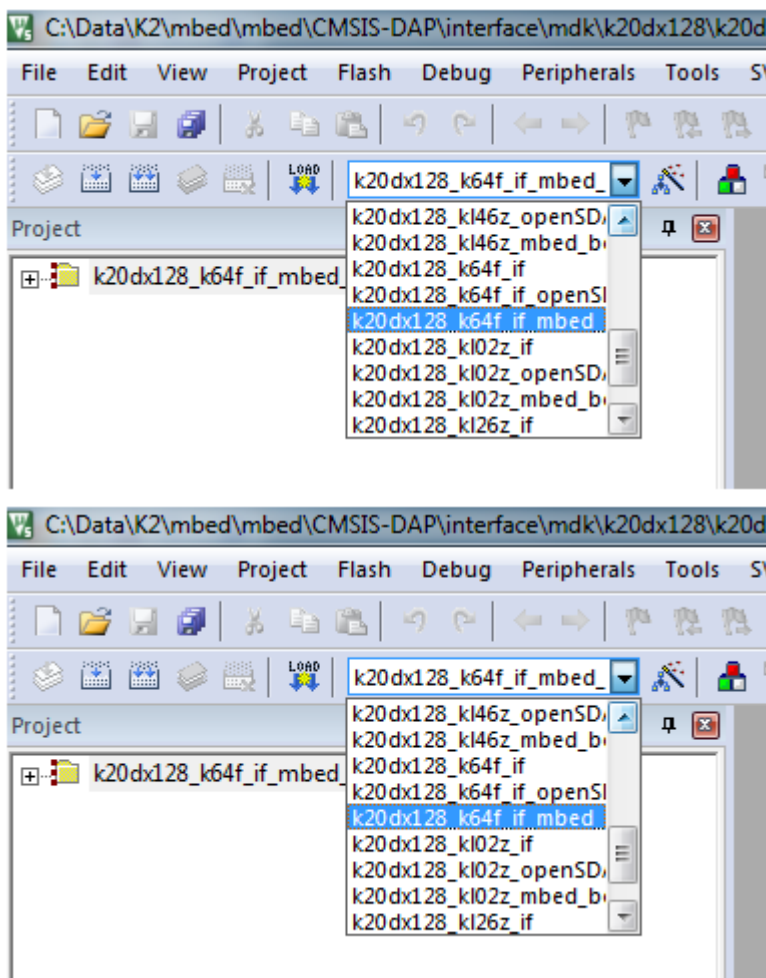
Data library				
CMSIS-DAP				
Name	Date modified	Type	Size	
.git	4/18/2014 2:51 PM	File folder		
bootloader	4/18/2014 2:51 PM	File folder		
interface	4/18/2014 2:51 PM	File folder		
shared	4/18/2014 2:51 PM	File folder		
tools	4/18/2014 2:51 PM	File folder		
LICENSE	4/18/2014 2:51 PM	File	11 KB	
README.md	4/18/2014 2:51 PM	MD File	1 KB	
CMSIS-DAP - Shortcut	4/18/2014 2:51 PM	Shortcut	3 KB	
.gitignore	4/18/2014 2:51 PM	Text Document	1 KB	

Data library				
CMSIS-DAP				
Name	Date modified	Type	Size	
.git	4/18/2014 2:51 PM	File folder		
bootloader	4/18/2014 2:51 PM	File folder		
interface	4/18/2014 2:51 PM	File folder		
shared	4/18/2014 2:51 PM	File folder		
tools	4/18/2014 2:51 PM	File folder		
LICENSE	4/18/2014 2:51 PM	File	11 KB	
README.md	4/18/2014 2:51 PM	MD File	1 KB	
CMSIS-DAP - Shortcut	4/18/2014 2:51 PM	Shortcut	3 KB	
.gitignore	4/18/2014 2:51 PM	Text Document	1 KB	

This repository contains the files for both the bootloader and the CMSIS-DAP debug interface application. We will concentrate on the interface application at the moment.

Open up Keil MDK, and open up the project file located at **\\CMSIS-DAP\\interface\\mdk\\k20dx128\\k20dx128\_interface.uvproj**

In the project configuration drop-down box, you will notice there are a lot of options. Since different chips may have slightly different flash programming algorithms, there is a target for each specific evaluation board. In this case, we will be building for the FRDM-K64F board. Scroll down until you get to that selection:



Notice there are three options for the K64:

- **k20dx128\_k64f\_if:** Used for debugging the CMSIS-DAP application with Keil. Code starts at address 0x0000\_0000
- **k20dx128\_k64\_if\_openSDA\_bootloader:** Creates a binary to drag-and-drop on the P&E developed bootloader (Original OpenSDA)
- **k20dx128\_k64\_if\_mbed\_bootloader:** Creates a binary to drag-and-drop onto the CMSIS-DAP/mbed developed bootloader (OpenSDAv2)



Since the FRDM-K64F comes with the OpenSDAv2 bootloader, we will use the 3<sup>rd</sup> option.

If we were building the mbed app for another Freedom board which had the original OpenSDA bootloader, we would choose the 2<sup>nd</sup> option instead.

Now click on the compile icon. You may get some errors

- If you get an error similar to the one shown below, make sure you have installed the Legacy\_pack for ARM as previously described earlier:

```
compiling RTX_Config.c...
```

```
..\..\Common\src\RTX_Config.c(184): error: #5: cannot open source input file
```

```
"RTX_lib.c": No such file or directory
```

and










```
compiling usb_config.c...
```

```
..\..\..\shared\USBStack\INC\usb_lib.c(18): error: #5: cannot open source input file "..\..\
```


```
\RL\USB\INC\usb.h": No such file or directory
```

- If you get an error regarding a missing version\_git.h file, make sure that Python 2.x and git are in your path. A Python build script fetches that file. It's called from the User tab in the project options, under "Run User Programs Before Build/Rebuild".
- If there is a warning about "invalid syntax" when running the Python script, make sure your using Python 2.x. Python 3.x will not work with the build script.


Now recompile again, and it should successfully compile. If you look now in **\CMSIS-DAP\interface\mdk\k20dx128** you will see a new **k20dx128\_k64f\_if\_mbed.bin** file

Name	Date modified	Type	Size
 Lst	4/18/2014 3:47 PM	File folder	
 Obj	4/18/2014 3:47 PM	File folder	
 k20dx128_interface.uvproj	4/18/2014 2:51 PM	µVision4 Project	752 KB
 <b>k20dx128_k64f_if_mbed.bin</b>	4/18/2014 3:55 PM	<b>BIN File</b>	<b>33 KB</b>
 k20dx128_interface_k20dx128_k64f_if_mbed_bootloader.d...	4/18/2014 3:55 PM	DEP File	52 KB
 k20dx128_interface_k20dx128_k64f_if_openSDA_bootload...	4/18/2014 3:46 PM	DEP File	36 KB
 k20dx128_interface_k20dx128_kl25z_if.dep	4/18/2014 3:39 PM	DEP File	32 KB
 kl25z_interface.uvopt.template	4/18/2014 2:51 PM	TEMPLATE File	51 KB
 .gitignore	4/18/2014 2:51 PM	Text Document	1 KB





Community



SIGN IN

Name	Date modified	Type	Size
Lst	4/18/2014 3:47 PM	File folder	
Obj	4/18/2014 3:47 PM	File folder	
k20dx128_interface.uvproj	4/18/2014 2:51 PM	µVision4 Project	752 KB
<b>k20dx128_k64f_if_mbed.bin</b>	4/18/2014 3:55 PM	BIN File	33 KB
k20dx128_interface_k20dx128_k64f_if_mbed_bootloader.d...	4/18/2014 3:55 PM	DEP File	52 KB
k20dx128_interface_k20dx128_k64f_if_openSDA_bootload...	4/18/2014 3:46 PM	DEP File	36 KB
k20dx128_interface_k20dx128_kl25z_if.dep	4/18/2014 3:39 PM	DEP File	32 KB
kl25z_interface.uvopt.template	4/18/2014 2:51 PM	TEMPLATE File	51 KB
.gitignore	4/18/2014 2:51 PM	Text Document	1 KB

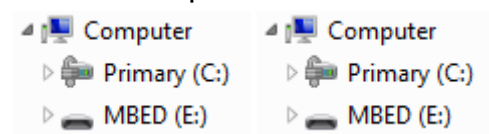
If you compiled the project for the OpenSDA bootloader, there would be a new **k20dx128\_k64f\_if\_openSDA.S19** file instead.

### Loading the CMSIS-DAP Debug Application:

Now take the Freedom board, press and hold the reset button as you plug in the USB cable. Then, drag-and-drop the .bin file (for OpenSDAv2) or .S19 file (for OpenSDA) into the BOOTLOADER drive that enumerated.



Perform a power cycle, and you should see a drive called “MBED” come up and you can start using the CMSIS-DAP debug interface, as well as drag-and-drop programming and virtual serial port as described earlier in this document.



## Appendix B: Building the CMSIS-DAP Bootloader

All Freedom boards already come with a bootloader pre-flashed onto the K20. But for those building their own boards that would like to use CMSIS-DAP, or those who would like to tinker with the bootloader, it possible to flash it to the Kinetis K20 device. Flashing the bootloader will require an external debugger, such as the Keil ULink programmer or Segger JLink.

Also note that the OpenSDA/PE Micro Bootloader cannot be erased! Due to the proprietary nature of the P&E firmware used by the original OpenSDA, it can only be

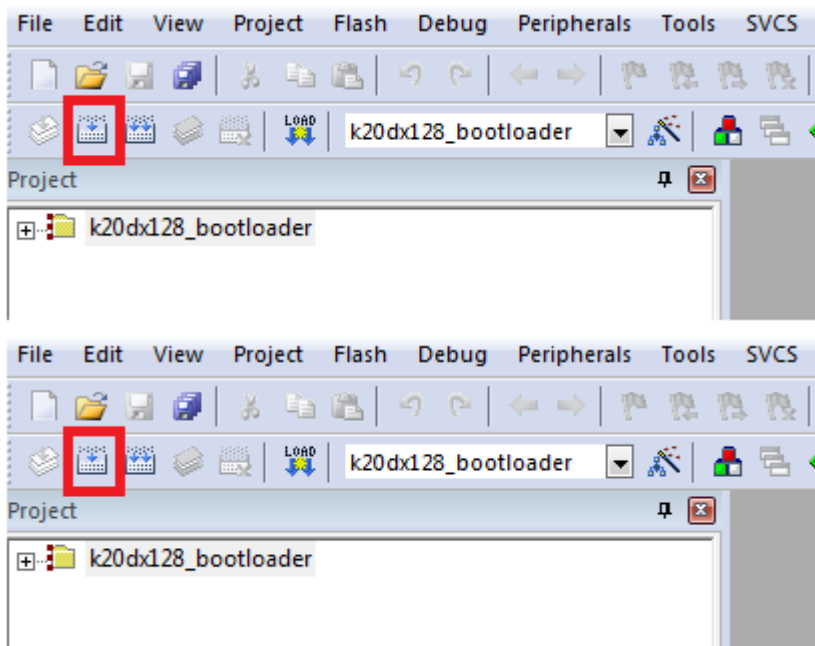
programmed at the board manufacturer and JTAG is disabled. So these instructions are applicable for boards with OpenSDAv2 only.

[SIGN IN](#)

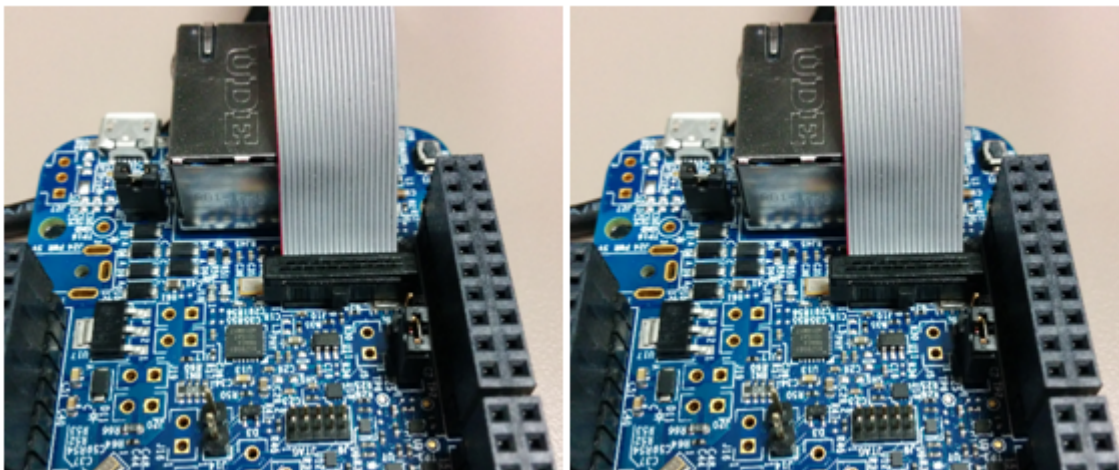
First, open up the bootloader project which is located at `\CMSIS-DAP\bootloader\mdk\k20dx128\k20dx128_bootloader.uvproj`

There is only one target available because all OpenSDAv2 boards will use the same bootloader firmware as the hardware circuitry is the same.

Click on the compile icon and it should compile successfully. If you see errors about a missing version\_git.h file, note that Python 2.x must be in the path to run a pre-build script which fetches that file.

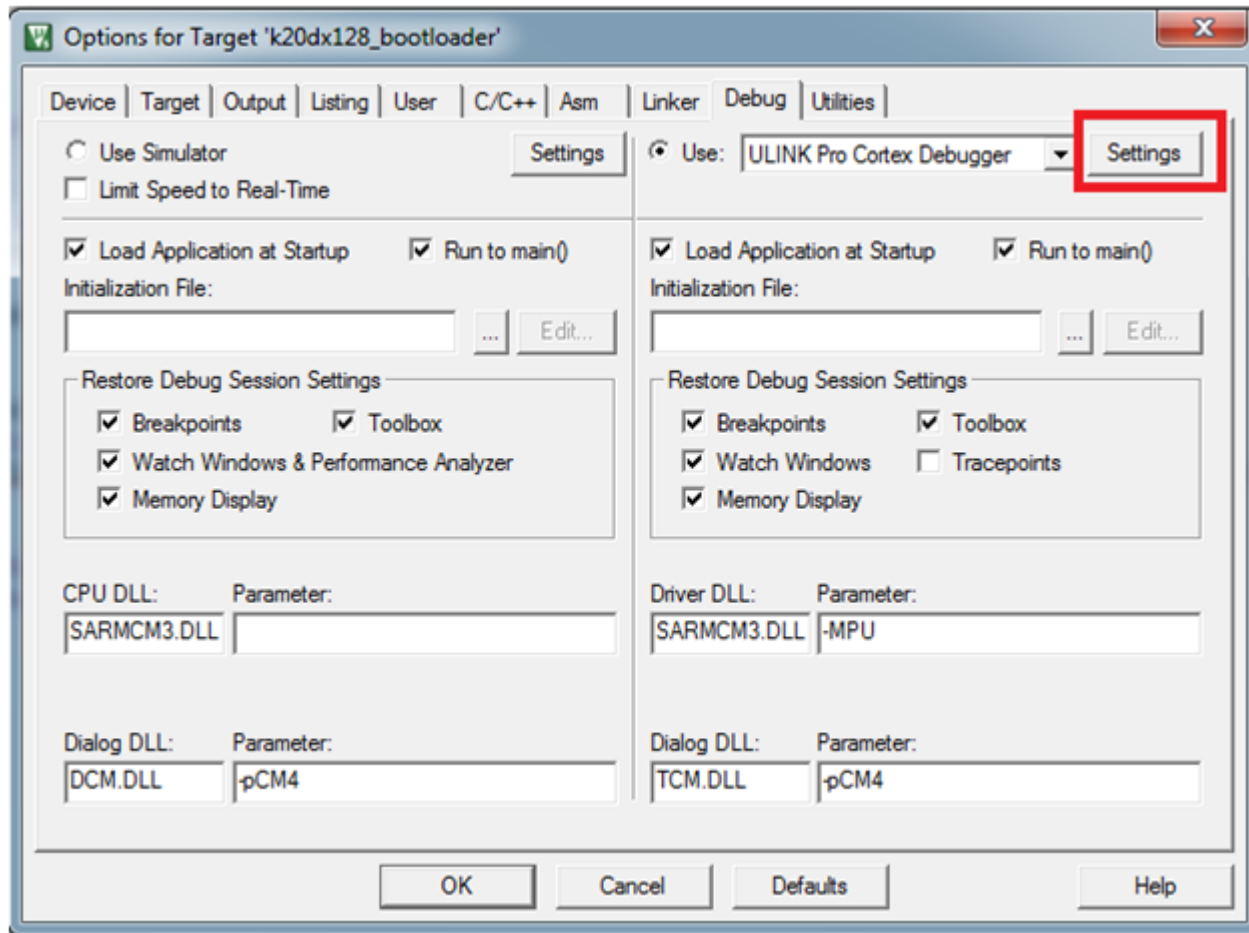


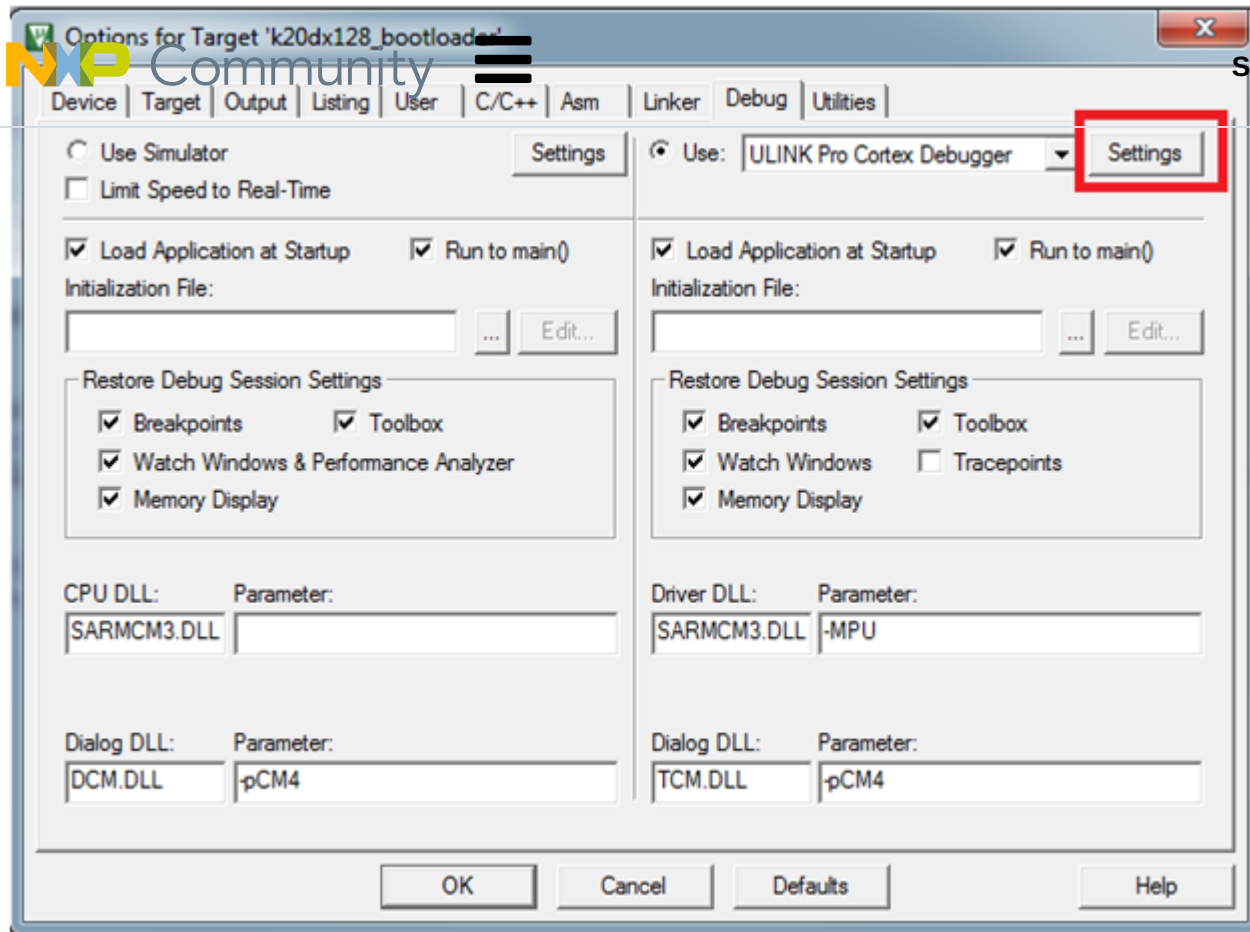
Now connect a Keil ULink to J10 and then insert a USB cable to provide power to J26. Note that if you have the 20-pin connector, you'll want to use the first 10 pins.



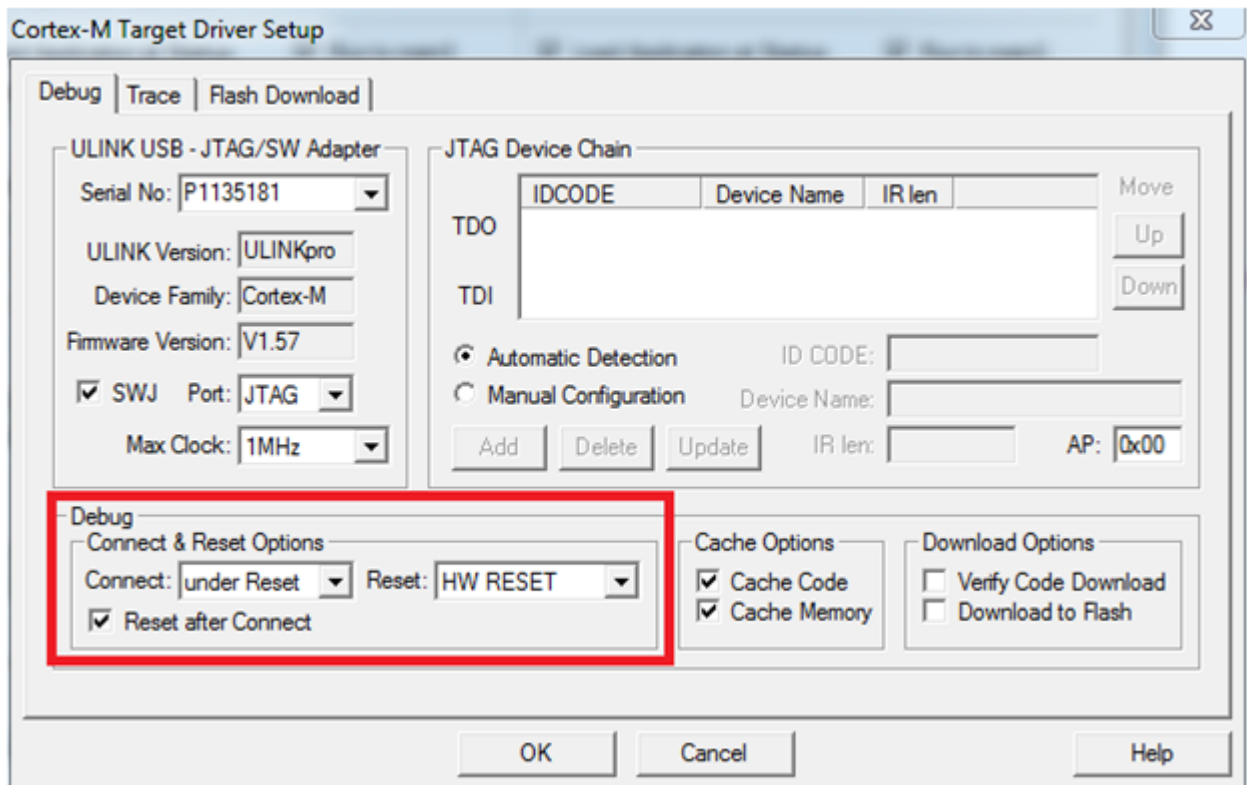
Then for Keil 5 you will need to change some debug options (CMSIS-DAP is built under

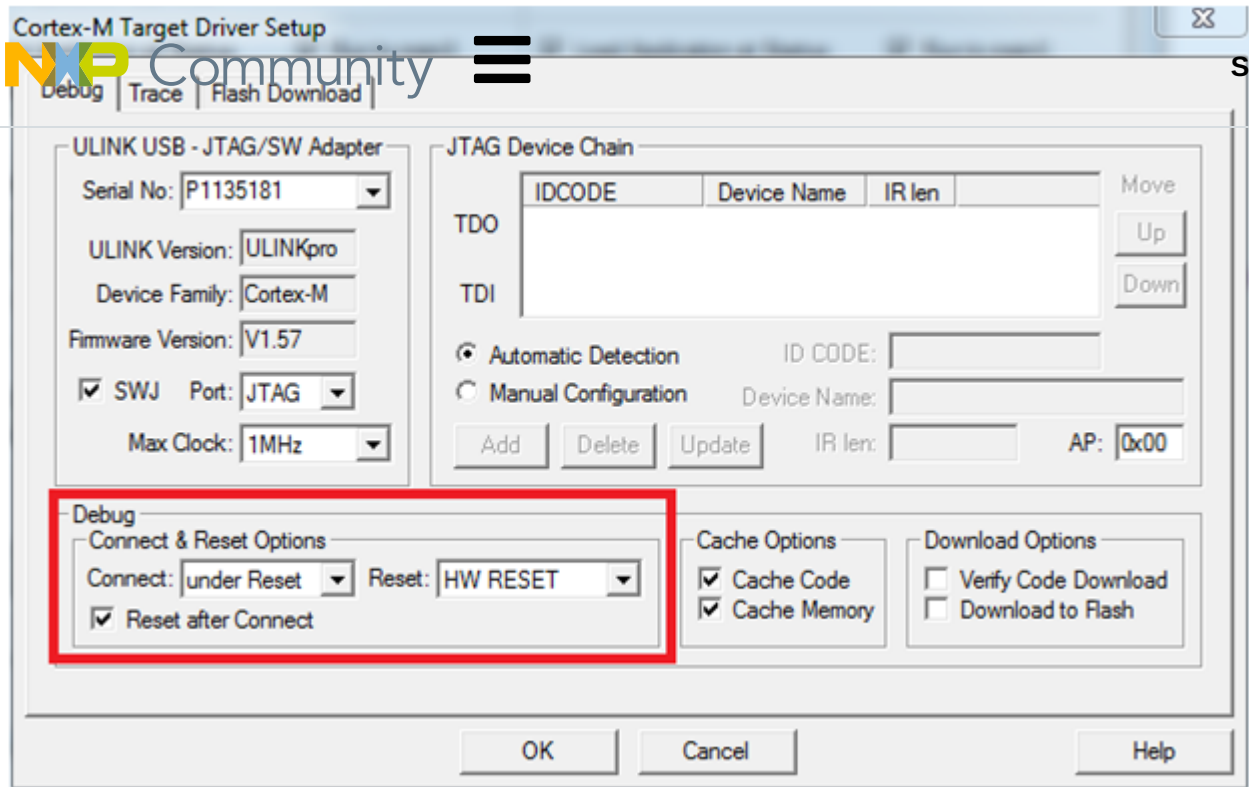
Right click on the bootloader project, and go to the Debug tab and next to ULINK Pro Cortex Debugger, click on Settings:



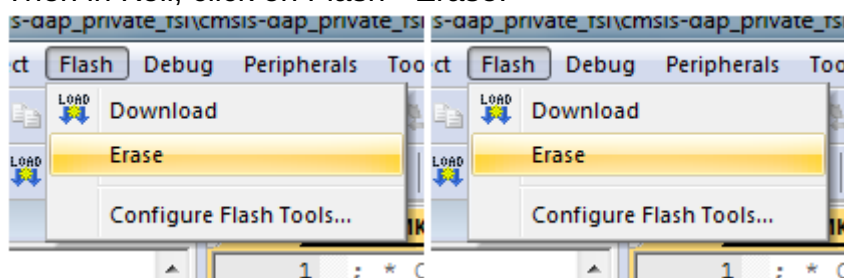


Then under “Cortex-M Target Driver Setup”, change the “Connect” drop down box to “under Reset” and “Reset” dropdown box to “HW RESET”. Hit OK to save the settings.

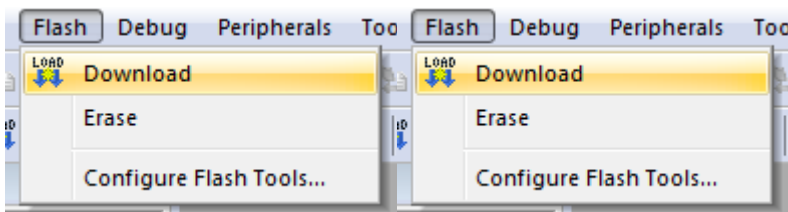




Then in Keil, click on Flash->Erase.



And then on Flash->Download.



If you get an "Invalid ROM Table" error when flashing the CMSIS-DAP bootloader, make sure you made the changes to the debugger settings listed above.

After some text scrolls by, you should see:

```
Erase Done.      Erase Done.
Programming Done. Programming Done.
Verify OK.       Verify OK.
```

Now power cycle while holding down the reset button, and you should see the bootloader drive come up. You'll then need to drag and drop the mbed application built earlier onto it. And that's all there is to it!

The binaries for the bootloader and CMSIS-DAP debug app for the FRDM-K64F board created in writing this guide are attached.

**Original Attachment has been moved to: [k20dx128\\_bootloader.axf.zip](#)**

**Original Attachment has been moved to: [k20dx128\\_k64f\\_mbed.bin.zip](#)**

Freedom Development Platform Kinetis K Series MCUs

cmsis-dap

frdm freedom frm-k64f kinetis openocd opensda opensdav2



14 Kudos

Was this article helpful?

YES

NO

SHARE

## COMMENTS



china-imm-fae-b

06-19-2014 08:59 PM

thanks.

opensda和opensda v2在硬件原理图上也是不同的。



JonathanHess

07-21-2014 07:20 AM

Thanks for this informative posting!

If there is no difference between the circuit for OpenSDA and OpenSDAv2, can we just load OpenSDAv2 on any OpenSDA system... say a TWR-KW2x. That way we pick up a more mainline ARM CMSIS-DAP debugging protocol in our toolchains. It would also make things more common between the different Freescale development boards and kits.



anthony\_huereca

07-21-2014 11:05 AM

You can load an OpenSDAv2 application onto an OpenSDA board. The one caveat is that the application needs to support the flash algorithm for your particular target device (KW2x in your case). That's why there are multiple targets for the OpenSDAv2 project file in Keil, so that the application can be created for a specific flash algorithm for a specific device. The KW2x isn't explicitly supported in the public OpenSDAv2 repository, but you should be able to use one of the other available targets, or make some small modifications in the flash algorithm files, to get it to work with your KW2x.

You won't be able to load the OpenSDAv2 bootloader on an already existing TWR-KW2x board programmed with the OpenSDA bootloader already due to flash protection in the bootloader. However because it's the application that determines the debug interface, you can still standardize on CMSIS-DAP by using the CMSIS-DAP application.

There's also some updates coming to the public repository, including a fix for a bug where you can only program an application once when using the OpenSDAv2 bootloader. Attempts to change the OpenSDAv2 application will fail after that initial download. This doesn't affect boards with OpenSDAv2 that come directly from Freescale since it uses a slightly different code-base, but the bootloader binary I uploaded has the bug. The fix is to modify the `flash_hal_erase_sector()` function in `flash_hal.c` to pass "SECTOR\_SIZE" instead of "1" to the two function calls.





JonathanHess

07-21-2014 04:04 PM

You won't be able to load the OpenSDAv2 bootloader on an already existing TWR-KW2x board programmed with the OpenSDA bootloader already due to flash protection in the boot loader.

Looking at chapter 8 of:

[http://cache.freescale.com/files/32bit/doc/ref\\_manual/K20P48M50SF0RM.pdf?fasp=1&WT\\_TYPE=Reference%20...](http://cache.freescale.com/files/32bit/doc/ref_manual/K20P48M50SF0RM.pdf?fasp=1&WT_TYPE=Reference%20...)

So not only was flash security enabled but mass erase was also disabled?  
:smileysad:

However because it's the application that determines the debug interface, you can still standardize on CMSIS-DAP by using the CMSIS-DAP application.

After re-reading this and your original posting — particularly Appendix A, I finally understand what this means. Hooray!



anthony\_huereca

07-24-2014 02:27 PM

Yes, as I understand it, mass erase is disabled with OpenSDAv1. But glad you got the application part figured out!

Also the flash fix I mentioned in my previous post is now in the public repository.





ConstYu

08-08-2014 09:16 PM

Hi Anthony,

Very good sharing, it make me clear about the feature of different OpenSDA version, but i have one doubt about the describation "make sure that Python 2.x and **git are in your path**", what's mean of " make sure git are in your path", Actully i have install Python2.7, but it remain to report missing version\_git.h file, I googled the key word version\_git.h, but find no effective solution.



anthony\_huereca

08-09-2014 12:10 PM

You need to install Git on your computer: <http://git-scm.com/>

The Keil project is setup so that when you compile, a Python script is executed, and inside that script, it uses Git to download the latest version of that file.



1 Kudo



kai\_liu

09-02-2014 06:27 PM

Hi, Anthony,

Is it true that we can load OpenSDA v2.1 application into OpenSDA v1 debugger like KL25Z/KL46, after converting bin to s19 ?

OpenSDA v1 application starts from 0x8000.

---

OpenSDA v2 application starts from 0x5000,

OpenSDA v2.1 changes back to 0x8000 in K22F board.

### Hardware compatibilities

V1 SDA\_SWD\_EN/SDA\_SWD\_OE are active low to 74LVC125ADB's nOE pin

V2 SDA\_SWD\_EN/SDA\_SWD\_OE are active low to 74LVCH1T45 (when DIR is high, B=A as output) and NLX2G14.

V1 SDA\_SPI0\_SIN <- SWD\_DIO\_TGTMCU (controlled by EN)

V2 SDA\_SPI0\_SIN <- SWD\_DIO\_TGTMCU (always connected)

P5V\_USB\_SENSE is moved to P30/PTD5.

Thanks.

It bothers me a lot, since I am going to release a OpenSDA v1 based multi-tool including debugger and other lab tools.



anthony\_huereca

09-03-2014 04:18 PM

Since the start address is now the same, an app created for OpenSDAv1 would run on a board with OpenSDAv2.1. However there are a number of caveats:

1) As you state, you would need to modify the OpenSDA app from S19 to a binary file. Note that .SDA files which are commonly used for OpenSDAv1 apps are encrypted and can't be converted, and thus can only be used with OpenSDAv1 bootloaders.

2) There may be some board specific code in the apps, so simply taking an OpenSDAv1 app for one board and trying to use it on a completely different chip/board running OpenSDAv2 may not work. The MSD feature will not work when switching target devices because it's specific to that particular device's flash algorithm.

3) Any hardware differences between boards are NOT due to OpenSDAv2. ~~Any~~ hardware differences are due to board or chip constraints or better practices as we found improvements. We've ran both OpenSDAv1 and OpenSDAv2 on the exact same board. That said, the app may have been tweaked for that particular board (see point #2) but the changes should be minor.

---



kai\_liu

09-17-2014 02:59 AM

This morning I tried to build it in Keil. I found that Python script stops me for git\_version.h. That means you have to install Python 2.7+ and Github. However, it seems the git\_version.h has not been included at all.

I have got my board back, I hope these firmware releases can run smoothly.

---



kai\_liu

09-17-2014 03:01 AM

You have to clone that source by git.

---



omolin

01-12-2015 08:27 AM

About this statement: "Also note that the OpenSDA/PE Micro Bootloader cannot be erased!"

I was able to flash OpenSDAv2 bootloader and then MBED CMSIS-DAP on a STM-KL25Z. It previously had the PE Micro bootloader but was erased by mistake (using the MDM-AP mass erase).

Because it's not reversible I have not tried this on any other boards so maybe this just works on certain older boards that may not have mass erase disabled.



felixkr

01-15-2015 09:06 AM

I am dealing with the python issue aswell and can't find the solution. Python seems to be installed correctly but i don't know how to include it into Keil. Anybody found a solution yet? Thanks.



anthony\_huereca

01-15-2015 10:39 AM

Hi Felix,

Two things to try:

- 1) Make sure you've downloaded the repository via git, using **"git clone <https://github.com/mbedmicro/CMSIS-DAP.git>"**
- 2) You can set the path to Python 2.7 explicitly in the project options. And you need to use Python 2.x. In the project options, go to the "User" tab, and under "Run User Programs Before Build/Rebuild" you'll see that "Run #1" is checked. By default it'll just have 'python' but you can put in the full path. So it will look something like this after your modification: "C:\programs\programming\python27\python ..\..\tools\pre\_build\_script.py"

-Anthony



wenxue

Community



04-14-2015 01:48 AM

**SIGN IN**

Hi Anthony,

Where to download the source code for OpensdaV2.1.

[ARMmbed/CMSIS-DAP · GitHub](#)

Is just the code for Opensdav2.0.

Thanks.

Best Regards,

Wenxue



kerryzhou

02-16-2016 10:49 PM

Hi Anthony,

Whether the opensda V2.1 is supporting the Window10 or not? Some customers find when they use the opensda v2.1, both the JLINK firmware or the CMSIS fireware can't work on their window10 PC.

Best Regards,

Jingjing



m4l490n

03-07-2016 09:22 PM

Hello

I'm trying to debug and I'm getting the following error:

**GNU ARM Eclipse 64-bits Open On-Chip Debugger 0.10.0-dev-00141-g09aeb96-dirty (2015-10-28-11:39)**

**Licensed under GNU GPL v2**

**For bug reports, read**

**<http://openocd.org/doc/doxygen/bugs.html>**

**embedded:startup.tcl:60: Error: Can't find kinetis.cfg**

**in procedure 'script'**

**at file "embedded:startup.tcl", line 60**

What am I missing?

Thanks!!



mmbds

04-20-2016 10:55 AM

Can you attach the bootloader in as a hex or bin format. I seem to be having a lot of issues converting to a bin file to program with the iar hardware tools.



jonbuford

09-16-2016 04:50 AM

I'm doing a design for a board that will be mbed compatible to program a K22F

part. The K20 design that is on the FRDM board would work, but I'm looking at if there are options that would be cost reduced over this. I've not looked at the source yet for OpenSDA 2, but I figured that someone here might be able to give me a quick answer if it should be portable to run on something like a KL25Z, or does it need the resources that the K20 has other than the USB? For my current design, I think I'll stick with the K20, but it would be nice to know that I have options for cost reduction in the longer term.

I guess the other question would be if we could reasonably use a K20 with smaller flash than the 128K that is used. Looking at the binary size of the current V2.1 firmware, it is only 20KB, is there a good reason for not using one of the 32KB parts (I know there might be some additional features and space requirements in the future.)?



jonbuford

09-17-2016 03:56 AM

Looking into it some more, for the same cost, the MK22FN128VLH10 would have the same flash as the original K20, more RAM and more IO available for less or equal to the 32K K20 parts. Given that they are pretty similar parts, I'm guessing that I'll be able to port the existing firmware over pretty easily.



ruudsiebieski

11-27-2018 02:26 AM

Maybe silly question; can OpenSDA v2 be used together with a PowerCore target (e200, MPC5748G) ?



1 Kudo

how to download the bootloader to MK20DX128? WHAT tools?

---

## Version history

**Last update:** 05-01-2014 12:44 PM

**Updated by:**  anthony\_huereca



[ABOUT NXP](#) [CAREERS](#) [INVESTORS](#) [MEDIA](#) [CONTACT](#) [SUBSCRIBE](#)



[Privacy](#) | [Terms of Use](#) | [Terms of Sale](#) | [Slavery and Human Trafficking Statement](#) | [Accessibility](#)

©2006-2023 NXP Semiconductors. All rights reserved.