Quick Start Guide – Draft jwestmoreland

This is a quick start guide for adding the Segger[1] JLink JTAG/SWD I/F to the Programmer's List that is supported with the Arduino[2] IDE. At the time of this writing, the release version for the IDE is version 1.8.13.

Disclaimer: This is a WIP (Work In Progress and no claims of success, fitness, guarantees or workmanship are extended) – and the procedure covered here will result in the bootloader firmware being (erased and) written to the target device. If you are not familiar with this process previously, you must proceed with caution. Under normal circumstances, it won't be necessary to replace the bootloader on your device unless some major enhancement(s) are enabled and the vendor – Arduino – provides an associated update tool and path. You are cautioned it is unlikely but possible following the procedure here could 'brick' your device. OK, now that's out of the way, let's continue.

Before using the JLink with OpenOCD[3] to run the Burn Bootloader app – please make sure you are familiar with the following wiki page:

https://wiki.segger.com/OpenOCD

The Arduino Platform specification is here:

https://arduino.github.io/arduino-cli/latest/platform-specification/

The three files that need to be modified are:

boards.txt

platform.txt and

programmers.txt

A local version of boards.txt and platform.txt can be created to override the default settings by creating:

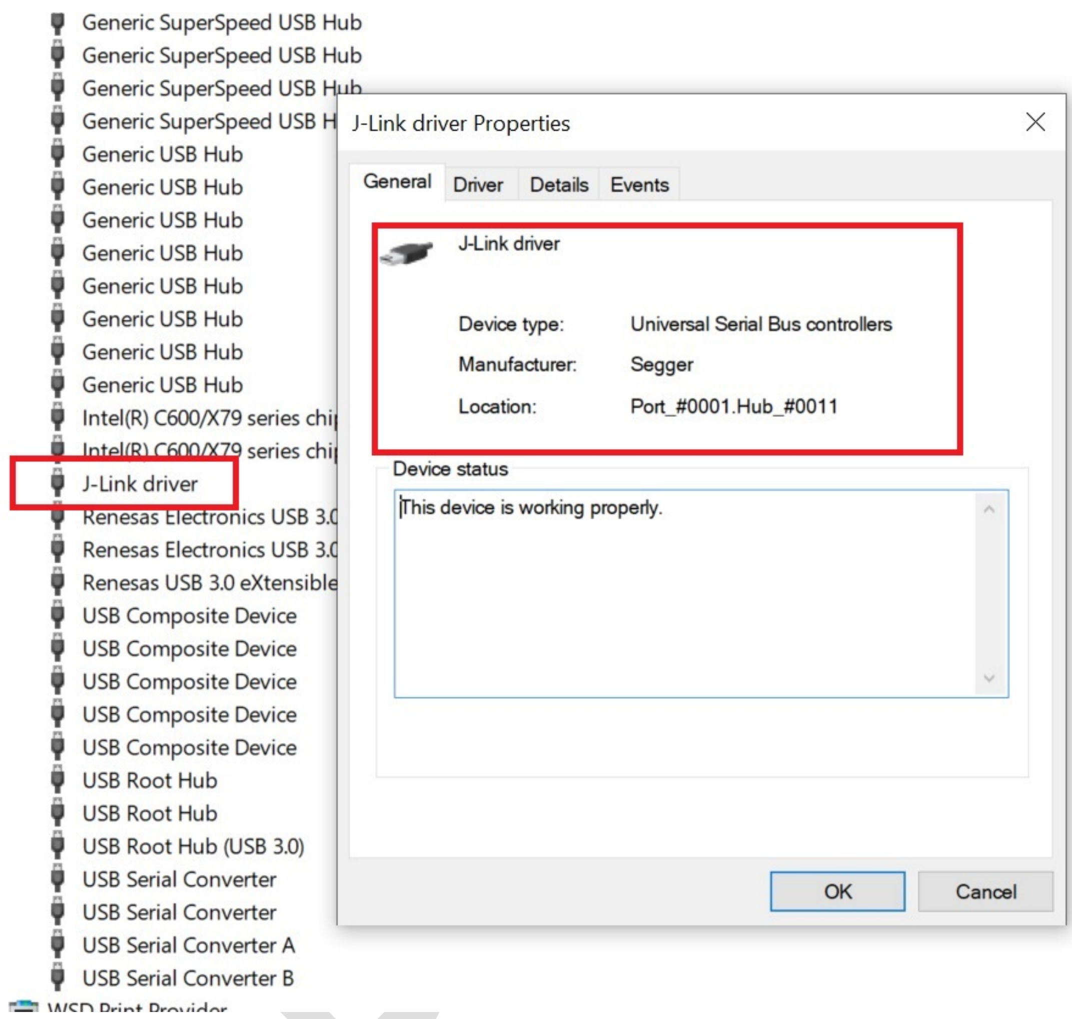
boards.local.txt and platform.local.txt.

The current directory path of the file location should follow a format like (on Win10):

C:\Users\<$userName>\AppData\Local\Arduino15\packages\arduino-beta\hardware\mbed\1.2.1

The modified files can be currently found here: 664f83a

The Zadig[4] tool is a utility to replace the JLink driver with WinUSB:



Generic SuperSpeed USB Hub
Generic SuperSpeed USB Hub
Generic SuperSpeed USB Hub
Generic SuperSpeed USB H
Generic USB Hub
Generic USB Hub
Generic USB Hub
Generic USB Hub
Generic USB Hub
Generic USB Hub
Generic USB Hub
Generic USB Hub
Intel(R) C600/X79 series chi
Intel(R) C600/X79 series chi
J-Link driver
Renesas Electronics USB 3.0
Renesas Electronics USB 3.0
Renesas USB 3.0 eXtensible
USB Composite Device
USB Composite Device
USB Composite Device
USB Composite Device
USB Composite Device
USB Root Hub
USB Root Hub
USB Root Hub (USB 3.0)
USB Serial Converter
USB Serial Converter
USB Serial Converter A
USB Serial Converter B
WSD Print Provider

J-Link driver Properties

General | Driver | Details | Events

J-Link driver

Device type:	Universal Serial Bus controllers
Manufacturer:	Segger
Location:	Port_#0001.Hub_#0011

Device status

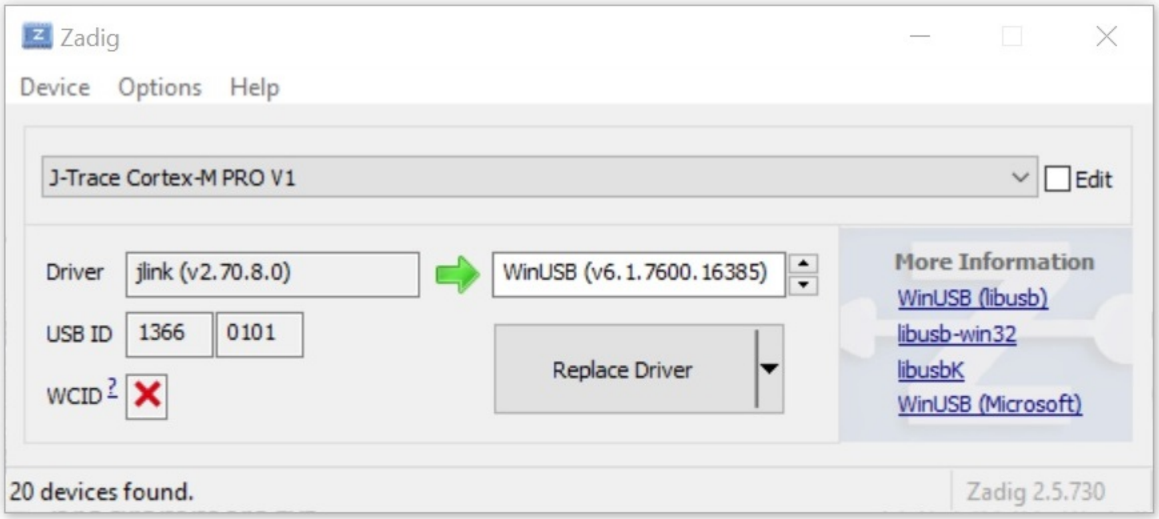
This device is working properly.

Figure 1: JLink driver to be replaced by the Zadig utility (example under win10).

The Zadig tool can be downloaded here: https://zadig.akeo.ie/

Open the Zadig utility, select Options from the main menu tabs and press list all devices:

Select your JLink device from the Drop-Down menu in the Zadig device list.

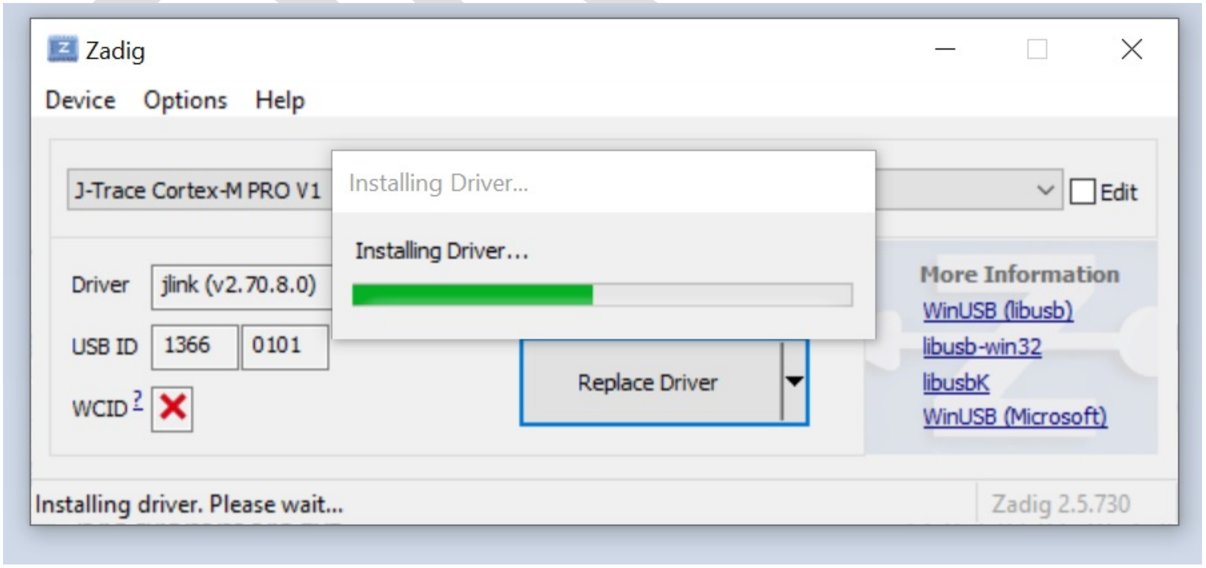


Zadig
Device Options Help
J-Trace Cortex-M PRO V1 ☐ Edit
Driver: jlink (v2.70.8.0) → WinUSB (v6.1.7600.16385)
USB ID: 1366 0101
WCID ?: ✗
Replace Driver
More Information: WinUSB (libusb), libusb-win32, libusbK, WinUSB (Microsoft)
20 devices found.
Zadig 2.5.730

Figure 2: Segger JLink Tool Selected showing jlink driver.

If necessary, arrow up/down in the small driver selection window to select WinUSB[5].

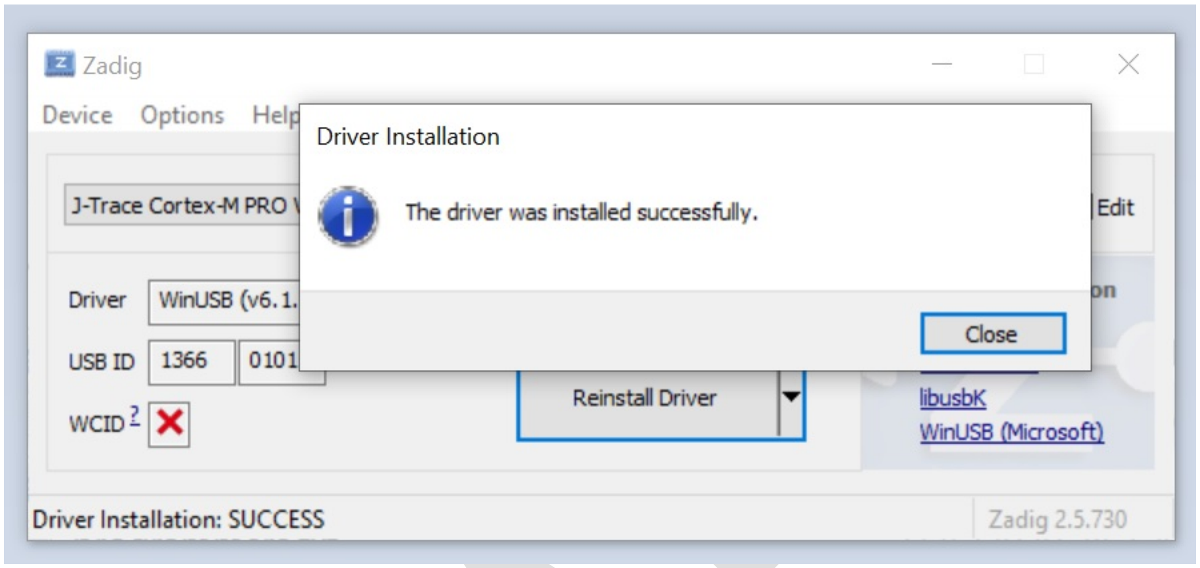
Now, press Replace Driver to replace the jlink driver with WinUSB so it will work with the OpenOCD interface.



Zadig
Device Options Help
J-Trace Cortex-M PRO V1 ☐ Edit
Installing Driver...
Installing Driver...
Driver: jlink (v2.70.8.0)
USB ID: 1366 0101
WCID ?: ✗
Replace Driver
More Information: WinUSB (libusb), libusb-win32, libusbK, WinUSB (Microsoft)
Installing driver. Please wait...
Zadig 2.5.730

Figure 3: WinUSB driver installing for the JLink device.

You should see a screen like the following that shows the driver successfully installed:

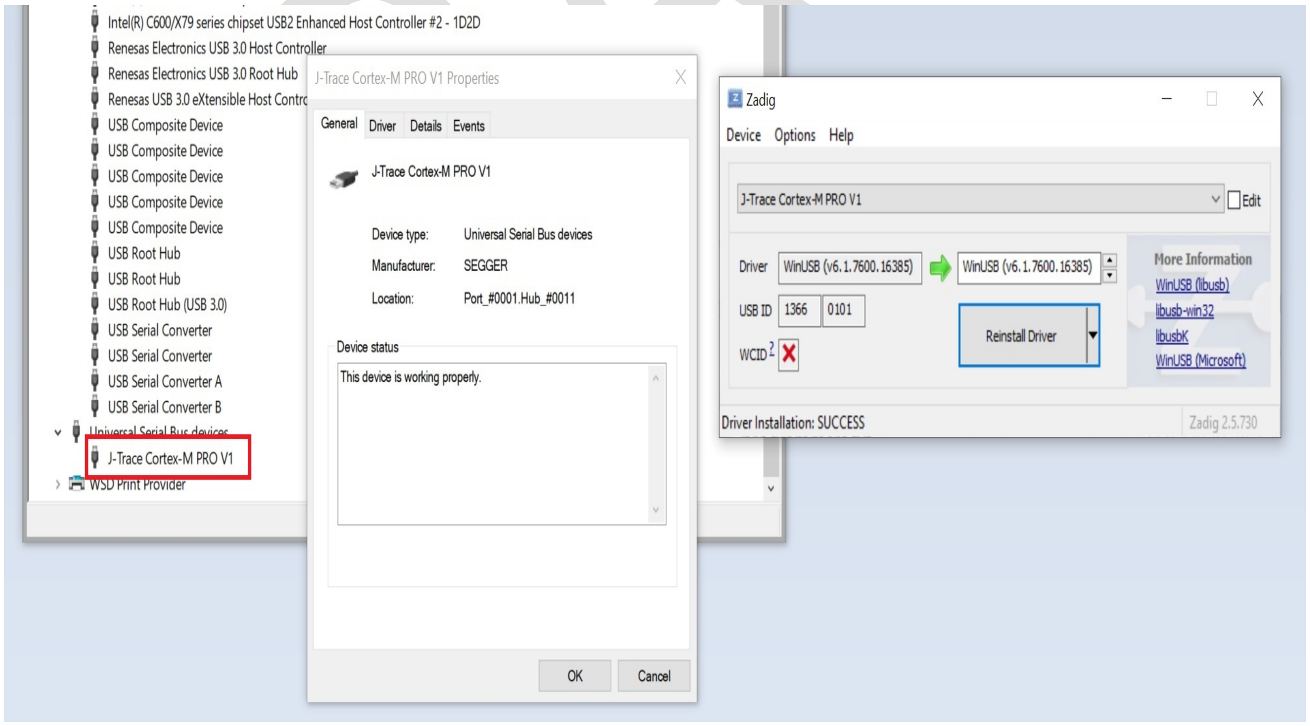


Driver Installation

The driver was installed successfully.

Driver Installation: SUCCESS

Figure 4: Zadig tool showing WinUSB driver has successfully installed.



J-Trace Cortex-M PRO V1

Device type: Universal Serial Bus devices

Manufacturer: SEGGER

Location: Port_#0001.Hub_#0011

Device status

This device is working properly.

J-Trace Cortex-M PRO V1

Driver: WinUSB (v6.1.7600.16385) → WinUSB (v6.1.7600.16385)

USB ID: 1366 0101

WCID

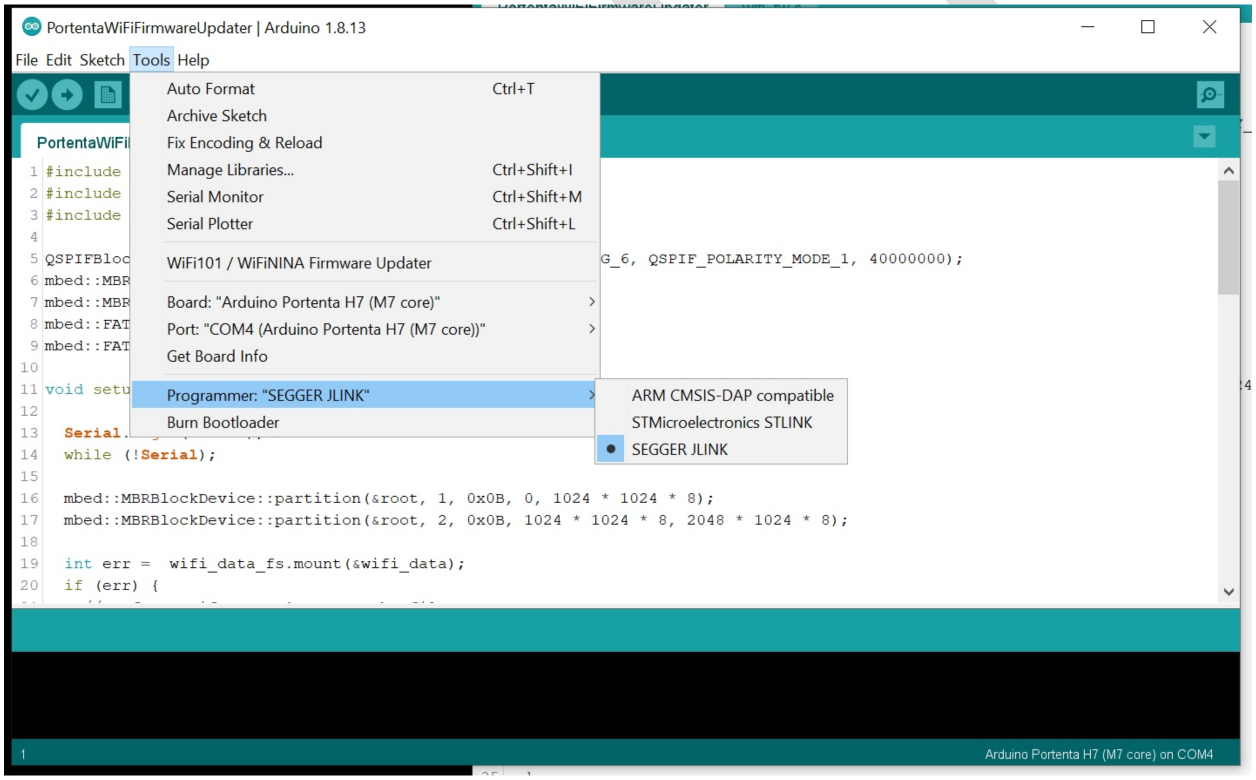
Driver Installation: SUCCESS

Figure 5: Successful install of the WinUSB driver making it ready to use with OpenOCD.

The Zadig utility can be closed now.

The Burn Bootloader command will program the preassigned firmware build assigned by Arduino to target using OpenOCD and a programming interface such as the Segger JLink.

The following screen grab shows an example of the Segger JLink being ready to program the bootloader for the Portenta H7. The particular sketch chosen does not come into play regarding the bootloader update for the main processor(s).


PortentaWiFiFirmwareUpdater | Arduino 1.8.13
File Edit Sketch Tools Help
PortentaWiFi
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Portenta H7 (M7 core)"
Port: "COM4 (Arduino Portenta H7 (M7 core))"
Get Board Info
Programmer: "SEGGER JLINK"
Burn Bootloader
ARM CMSIS-DAP compatible
STMicroelectronics STLINK
● SEGGER JLINK
Arduino Portenta H7 (M7 core) on COM4

```
#include
#include
#include

QSPIFBloc                                              G_6, QSPIF_POLARITY_MODE_1, 40000000);
mbed::MBR
mbed::MBR
mbed::FAT
mbed::FAT

void setu

  Serial.
  while (!Serial);

  mbed::MBRBlockDevice::partition(&root, 1, 0x0B, 0, 1024 * 1024 * 8);
  mbed::MBRBlockDevice::partition(&root, 2, 0x0B, 1024 * 1024 * 8, 2048 * 1024 * 8);

  int err =  wifi_data_fs.mount(&wifi_data);
  if (err) {
```

Figure 7: Once files are installed, the SEGGER JLINK Option should show up as depicted above.

An example output of the tool once it connects successfully will be similar to the following – but could be different based on your particular JLink model, OpenOCD version, file paths, etc. – but it should successfully connect:

```
C:\tmp>c:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13/bin/openocd.exe -d2 -s C:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13/share/openocd/scripts/ -f interface/jlink.cfg -f target/stm32h7x_dual_bank.cfg -c "telnet_port disabled; init; stm32h7x option_write 0 0x01c 0xb86aaf0; program {C:\Users\john\AppData\Local\Arduino15\packages\arduino-beta\hardware\mbed\1.2.1/bootloaders\PORTENTA_H7\portentah7_bootloader_mbed_hs_v2.elf }; reset run; shutdown"

Open On-Chip Debugger 0.10.0+dev-g7c88e76a7-dirty (2020-07-06-09:59)

Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
debug_level: 2

Swd
Info : J-Trace PRO V1 Cortex-M compiled Jun  9 2020 13:39:24
Info : Hardware version: 1.00
Info : VTarget = 3.160 V
Info : clock speed 1800 kHz
Info : SWD DPIDR 0x6ba02477
Info : stm32h7x.cpu0: hardware has 8 breakpoints, 4 watchpoints
Info : starting gdb server for stm32h7x.cpu0 on 2331
Info : Listening on port 2331 for gdb connections
Info : Device: STM32H74x/75x
Info : flash size probed value 2048
Info : STM32H7 flash has dual banks
Info : Bank (0) size is 1024 kb, base address is 0x8000000
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0800ce18 msp: 0x24080000
** Programming Started **
Info : Padding image section 1 at 0x0801fcac with 20 bytes (bank write end alignment)
Warn : Adding extra erase range, 0x0801fcc0 .. 0x0801ffff
** Programming Finished **
shutdown command invoked
```

That’s just an example of a connection/programming session that was successful for illustration purposes.

It’s suggested to run a few preliminary connection tests before programming the bootloader – you can leave the program <file name> path out and run the command to verify connectivity.

Once you are successfully connected – you should be able to run the Burn Bootloader command.

At the time of the writing of this document, the default bootloader file is located here:

C:\Users\<$userName>\AppData\Local\Arduino15\packages\arduino-beta\hardware\mbed\1.2.1/bootloaders\PORTENTA_H7\portentah7_bootloader_mbed_hs_v2.elf

Appendix 1: Physically Connecting to the Debug Header for Portenta

Unfortunately, there's currently no 'easy' way to connect to the Debug/JTAG/SWD debug headers on the Portenta:

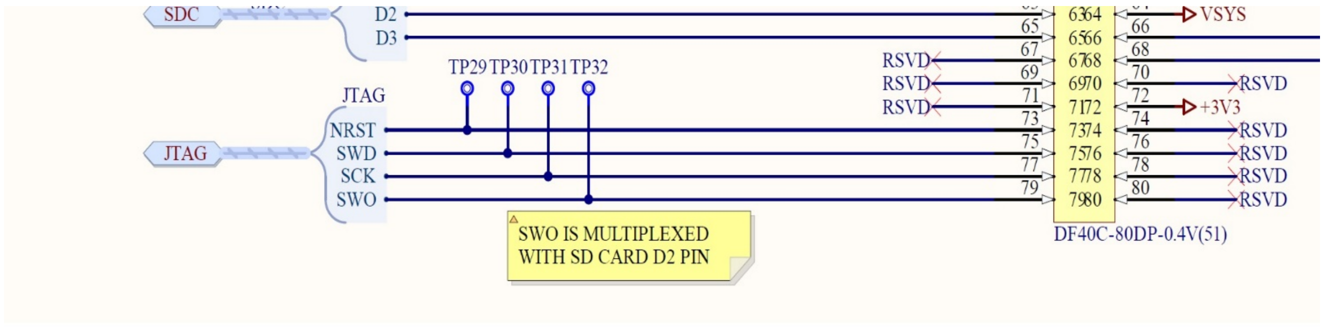


Figure A1-1: Portenta Debug Header on connector J1.

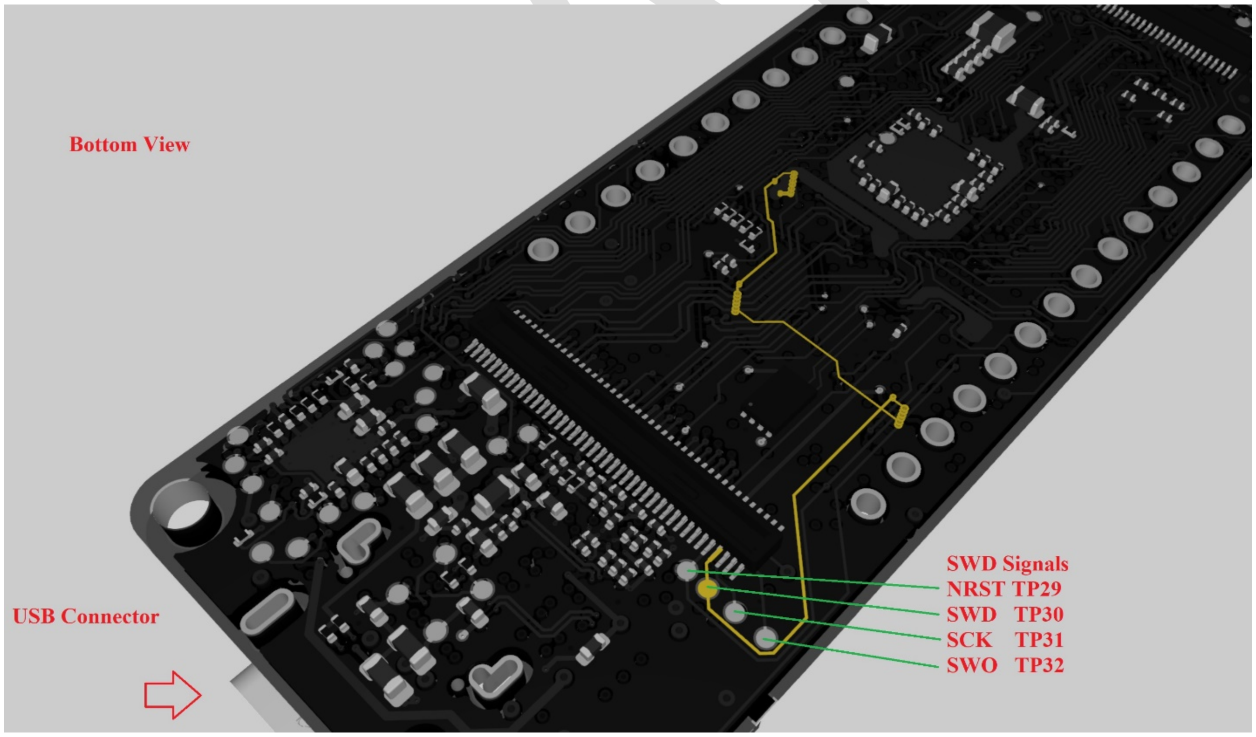


Figure A1-2: Portenta Debug Header – 3D View

Also, it is suggested to run a ground connection to the programming header since it'll help reduce noise during programming.

The Segger's SWD programming header is defined as:

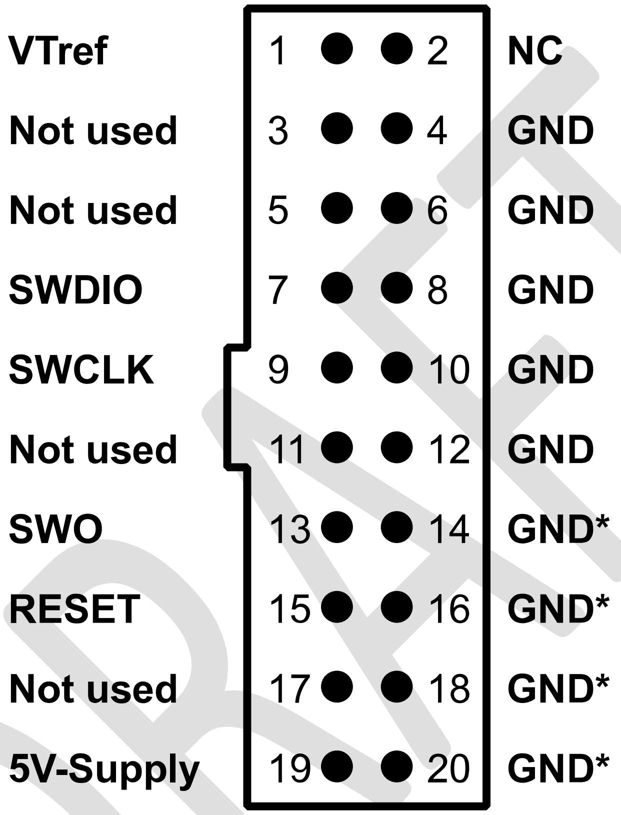


Figure A1-3: Segger JLink Programming Header SWD Pin Assignments

It is recommended to connect VTref to the +3V1 rail of the Portenta H7.

It is left as an exercise to the reader to build the programming harness of your choice and good luck!

Appendix 2: The Orange LED of DOOM (OLoD):

Let’s say you get adventurous and accidentally do something – like maybe programming a ‘wrong’ choice from this list:

portentah7_bootloader_mbed_fs.bin
portentah7_bootloader_mbed_fs.elf

portentah7_bootloader_mbed_hs.bin
portentah7_bootloader_mbed_hs.elf

portentah7_bootloader_mbed_hs_v2.bin
portentah7_bootloader_mbed_hs_v2.elf

And when the board reboots – DL2 – the Orange Led to the immediate right of the USB-C connector on the top side of the board, is constantly on:

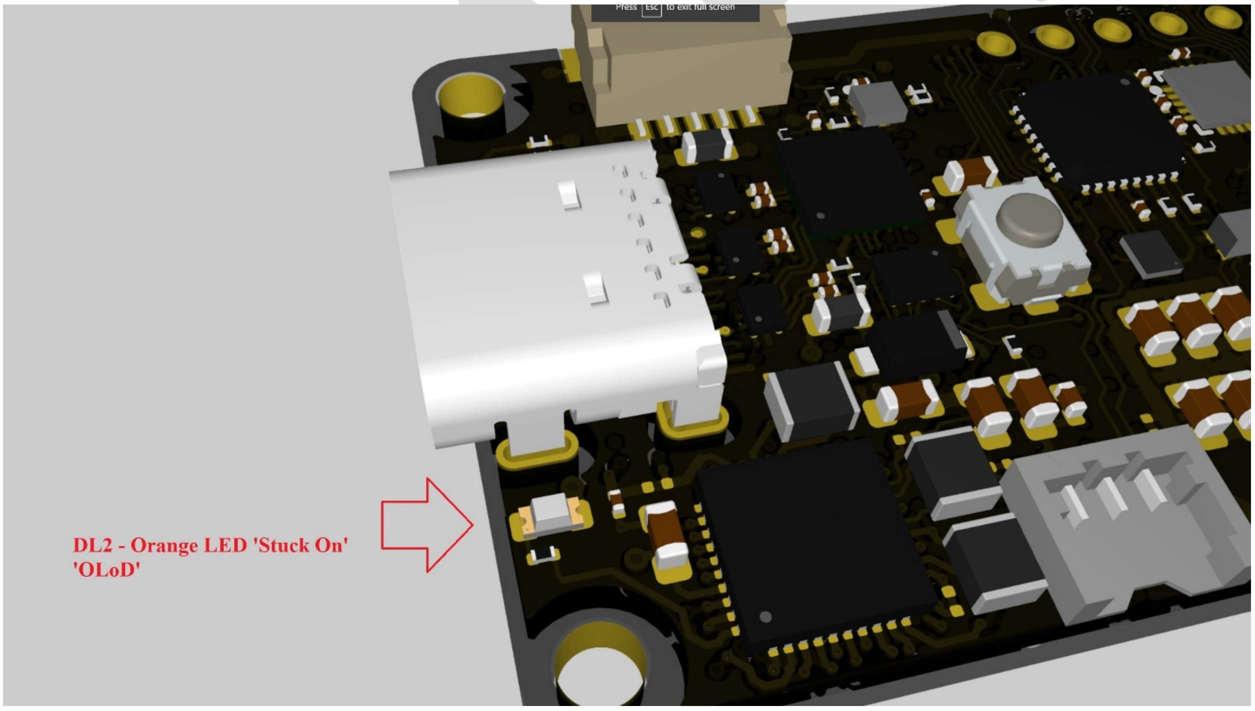


Figure A2-1: DL2 – The Orange LED above, will be ‘stuck on’; i.e. OLoD mode.

Symptoms:

USB won’t enumerate, pressing the reset button just results in DL2 remaining on.
Can’t connect with the IDE nor the programmer.

You will get a screen like this:

```
C:\tmp>c:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13\bin\openocd.exe -d2 -s C:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13\share\openocd\scripts\ -f interface/jlink.cfg -f target/stm32h7x_dual_bank.cfg -c "telnet_port disabled; tcl_port disabled; init; reset init; stm32h7x option_write 0 0x01c 0xb86aaf0; program {C:\Users\john\AppData\Local\Arduino15\packages\arduino-beta\hardware\mbed\1.2.1/bootloaders\PORTENTA_H7\portentah7_bootloader_mbed_hs_v2.elf }; reset run; shutdown"

Open On-Chip Debugger 0.10.0+dev-g7c88e76a7-dirty (2020-07-06-09:59)

Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
debug_level: 2

Swd
Info : J-Trace PRO V1 Cortex-M compiled Jun  9 2020 13:39:24
Info : Hardware version: 1.00
Info : VTarget = 3.111 V
Info : clock speed 1800 kHz
```

And OpenOCD will just exit and it'll appear like the target isn't connected.

I noticed periodically DL2 will blink – maybe once every 15 or 20 seconds or so, but otherwise it remains solidly lit.

One method to restore the bootloader is to put the Portenta on its own USB powered hub – with a power switch – it's very important to have a hub with a switch since it is necessary to time the Portenta's powering up with initiating programming the board.

This is the powered USB hub I used:

Figure A2-2: USB-C Compatible Hub by StarTech[6] Model #: HB30C5A2CST

It could take several attempts – but once you time the turn-on of the Portenta via the switch on the USB hub and pressing <Enter> to kick off the programming step, you will get something like the following:

```
C:\tmp>c:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13\bin\openocd.exe -d2 -s C:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13\share\openocd\scripts\ -f interface/jlink.cfg -f target/stm32h7x_dual_bank.cfg -c "telnet_port disabled; tcl_port disabled; init; reset init; stm32h7x option_write 0 0x01c 0xb86aaf0; program {C:\Users\john\AppData\Local\Arduino15\packages\arduino-beta\hardware\mbed\1.2.1/bootloaders\PORTENTA_H7\portentah7_bootloader_mbed_hs_v2.elf }; reset run; shutdown"

Open On-Chip Debugger 0.10.0+dev-g7c88e76a7-dirty (2020-07-06-09:59)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
debug_level: 2
```

```
swd
Info : J-Trace PRO V1 Cortex-M compiled Jun  9 2020 13:39:24
Info : Hardware version: 1.00
Info : VTarget = 3.095 V
Info : clock speed 1800 kHz
Info : SWD DPIDR 0x6ba02477
Info : stm32h7x.cpu0: hardware has 8 breakpoints, 4 watchpoints
Info : stm32h7x.cpu0: external reset detected
Info : starting gdb server for stm32h7x.cpu0 on 2331
Info : Listening on port 2331 for gdb connections
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0800c944 msp: 0x24080000
Info : Device: STM32H74x/75x
Info : flash size probed value 2048
Info : STM32H7 flash has dual banks
Info : Bank (0) size is 1024 kb, base address is 0x8000000
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0800c944 msp: 0x24080000
** Programming Started **
Info : Padding image section 1 at 0x0801fcac with 20 bytes (bank write end alignment)
Warn : Adding extra erase range, 0x0801fcc0 .. 0x0801ffff
** Programming Finished **
shutdown command invoked
```

You should see the board enumerate correctly now under USB and you can program it again via the IDE.

To Probe Further:

Don't give up trying to reprogram the board - if you get a screen like the following – it's a good sign and it means you're close to getting the timing correct with powering up the Portenta and kicking off the programming sequence:

```
C:\tmp>c:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13\bin\openocd.exe -d2 -s C:\Users\john\AppData\Local\Arduino15\packages\arduino\tools\openocd\0.10.0-arduino13\share\openocd\scripts\ -f interface/jlink.cfg -f target/stm32h7x_dual_bank.cfg -c "telnet_port disabled; tcl_port disabled; init; reset init; halt; stm32h7x option_write 0 0x01c 0xb86aaf0; program {C:\Users\john\AppData\Local\Arduino15\packages\arduino-beta\hardware\mbed\1.2.1/bootloaders\PORTENTA_H7\portentah7_bootloader_mbed_hs_v2.elf }; reset run; shutdown"
```

```
Open On-Chip Debugger 0.10.0+dev-g7c88e76a7-dirty (2020-07-06-09:59)
```

```
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
debug_level: 2

Swd
Info : J-Trace PRO V1 Cortex-M compiled Jun  9 2020 13:39:24
Info : Hardware version: 1.00
Info : VTarget = 3.098 V
Info : clock speed 1800 kHz
Info : SWD DPIDR 0x6ba02477
Info : stm32h7x.cpu0: hardware has 8 breakpoints, 4 watchpoints
Info : stm32h7x.cpu0: external reset detected
Error: Failed to write memory and, additionally, failed to find out where
Polling target stm32h7x.cpu0 failed, trying to reexamine
Examination failed, GDB will be halted. Polling again in 100ms
Error: Failed to write memory and, additionally, failed to find out where
Error executing event examine-end on target stm32h7x.cpu0:

Polling target stm32h7x.cpu0 failed, trying to reexamine
Examination failed, GDB will be halted. Polling again in 300ms
Info : starting gdb server for stm32h7x.cpu0 on 2331
Info : Listening on port 2331 for gdb connections
Info : AP write error, reset will not halt
TARGET: stm32h7x.cpu0 - Not halted
```

References:

1. Segger JLink: https://www.segger.com/products/debug-trace-probes/#j-link
2. Arduino IDE Download Page: https://www.arduino.cc/en/Main/Software
3. OpenOCD: http://openocd.org/
4. Zadig USB Driver Tool: https://zadig.akeo.ie/
5. WinUSB: https://docs.microsoft.com/en-us/windows-hardware/drivers/usbcon/winusb
6. StarTech Powered Hub: https://www.startech.com/Cards-Adapters/USB-3.0/Hubs/7-port-usb-c-hub~HB30C5A2CST

Revision History:

0p3: The Arduino CLI Platform specification changed to: The Arduino Platform specification as suggested by 'pert' online: https://forum.arduino.cc/index.php?topic=698354.0 .

Also, this revision history begun.