

MicroPython Firmware Flashing for STM32

การติดตั้งไฟล์เฟิร์มแวร์ของไมโครไพธอนสำหรับบอร์ดไมโครคอนโทรลเลอร์ STM32

ขั้นตอนการดำเนินการสำหรับบอร์ด STM32

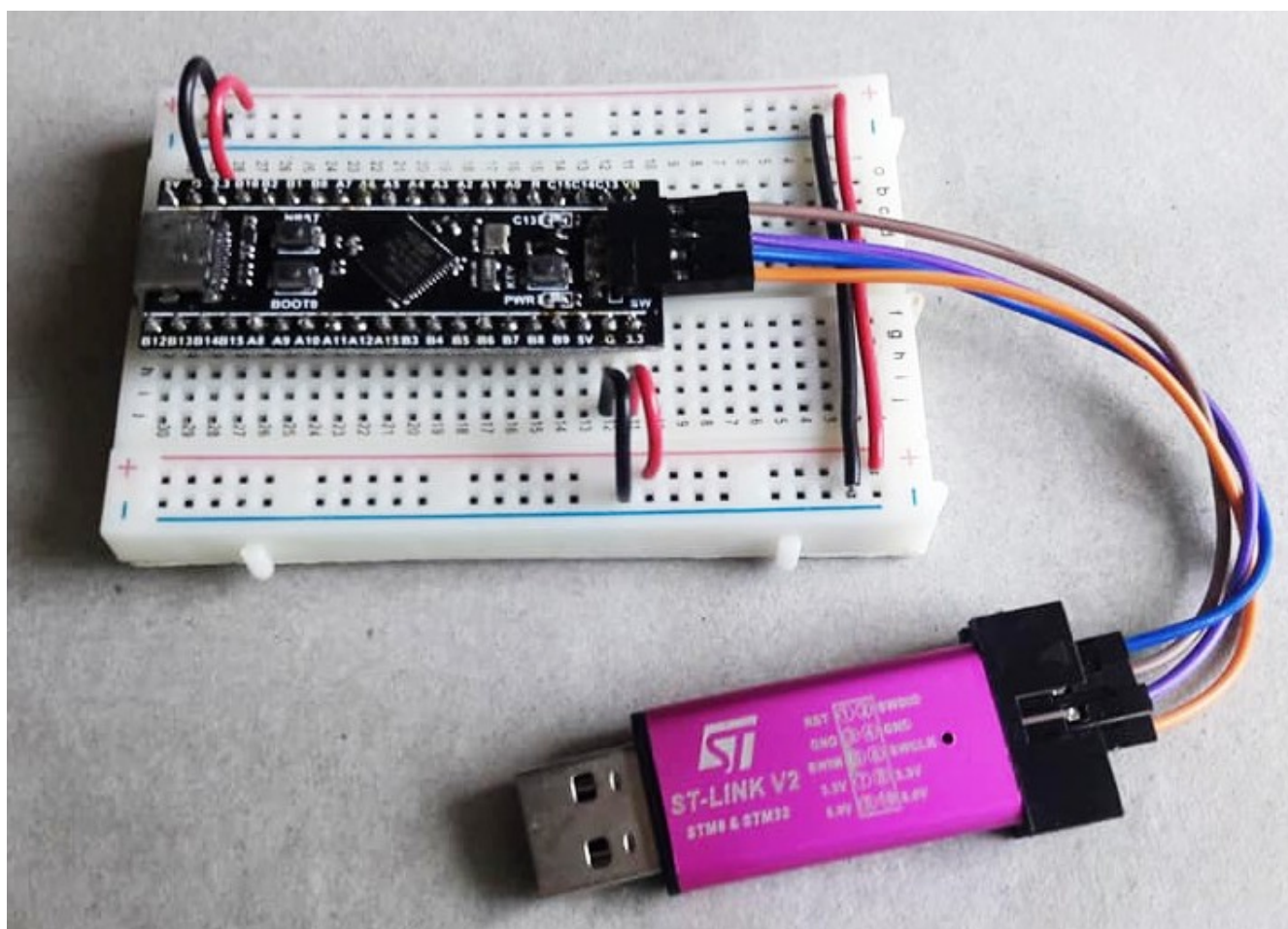
- ถ้าใช้บอร์ด **WeAct STM32F411CEU6 (Black Pill)** ให้ดาวน์โหลดและติดตั้งไฟล์ **MicroPython Firmware (pre-built binary file)** จาก **Github** ซึ่งมีประเภทของไฟล์ให้เลือก เช่น **.hex** หรือ **.dfu** และยังจำแนกตามรูปแบบการใช้หน่วยความจำ **Flash**
 - กรณีที่ใช้เฉพาะ **Internal Flash** (ไม่มี **External SPI Flash Chip**)
 - กรณีที่ใช้ **External SPI Flash 4MB**
 - กรณีที่ใช้ **External SPI Flash 8MB**
- ถ้าใช้บอร์ด **STM32 Nucleo** หรือ **Discovery** ให้ดาวน์โหลดไฟล์ **.dfu** ได้จาก <https://micropython.org/download/stm32/>
- **Windows**: ให้ดาวน์โหลดซอฟต์แวร์ของบริษัท **STMicroelectronics** แล้วติดตั้งในระบบให้พร้อมใช้งาน (มีขั้นตอนการริจิสเตอร์ผู้ใช้อย่างน้อยหนึ่งขั้นตอน จึงจะสามารถดาวน์โหลดไฟล์จากเว็บไซต์ของทางบริษัทได้) มีอยู่ 3 ตัวเลือกระหว่างนี้
 - โปรแกรม **STM32 ST-Link Utility** ถ้าใช้วิธีโปรแกรมผ่าน **SWD** (วิธีที่ 1) หรือ
 - โปรแกรม **DfuSe** ถ้าโปรแกรมด้วยวิธี **DFU** (วิธีที่ 2) หรือ
 - โปรแกรม **STM32CubeProgrammer** (ได้ทั้งวิธีที่ 1 และ 2) <= แนะนำให้เลือกใช้ตัวเลือกนี้ และโปรแกรมสามารถใช้ได้ทั้งกับ **Windows** และ **Linux**
 - โปรแกรม **dfu-util for Windows** เช่น **dfu-util-0.9-win64.zip**
- **Linux (Ubuntu, Raspbian OS)**: ให้ติดตั้งโปรแกรม เช่น **openocd**, **stlink-tools** และ **dfu-util** เพื่อเอาไว้ใช้งาน

การโปรแกรมด้วยวิธี SWD สำหรับ Windows

ถ้ามีอุปกรณ์ **ST-Link/V2** สำหรับการโปรแกรมด้วยวิธี **SWD** ก็สามารถใช้ไฟล์ **.hex** ได้เลย ในเครื่องคอมพิวเตอร์ (**Windows**) จะต้องมีการติดตั้งโปรแกรม **STM32 ST-Link Utility** ไว้แล้ว



รูปภาพ: อุปกรณ์ ST-Link V2 USB Dongle (Low-Cost)

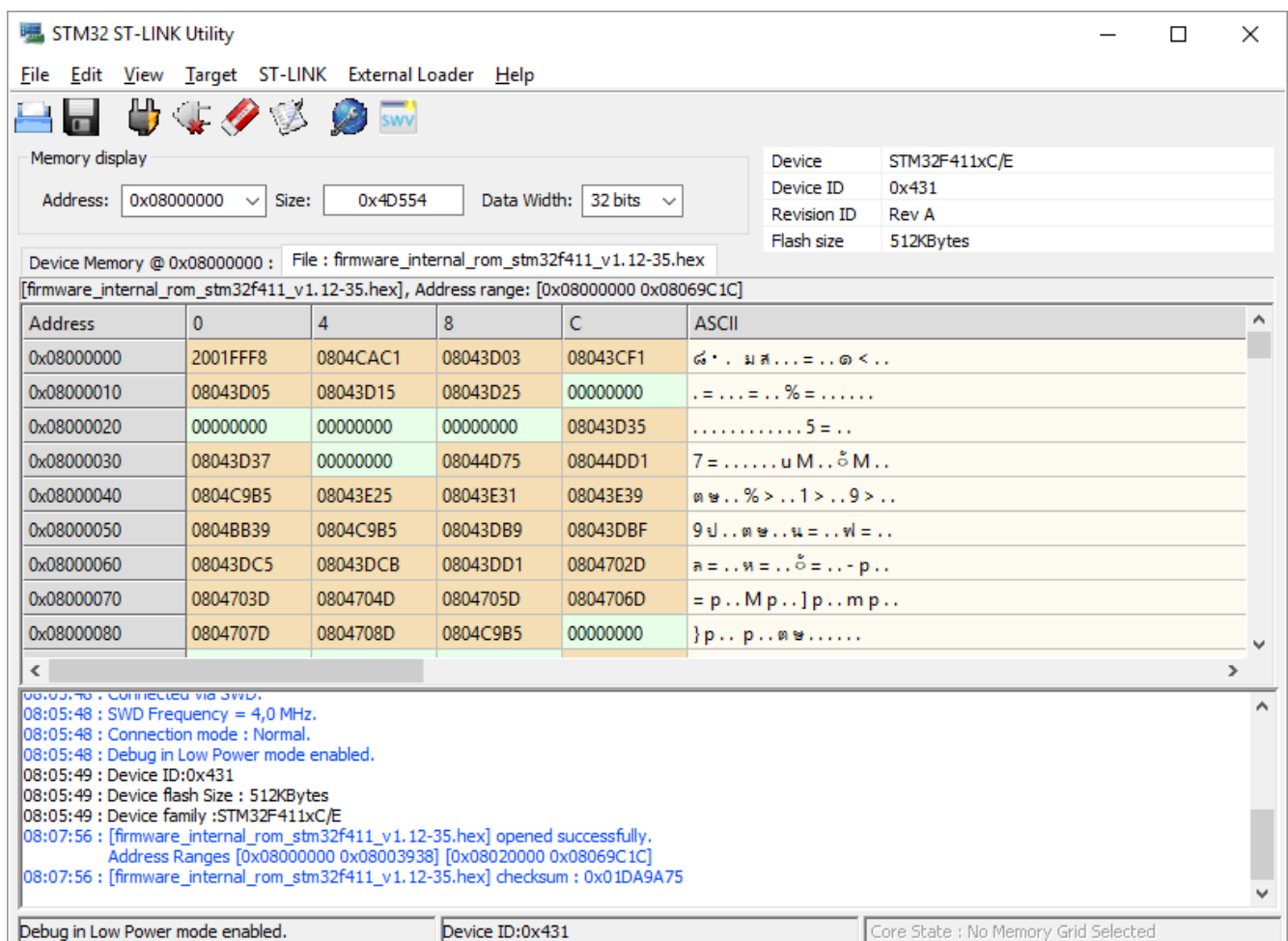


รูปภาพ: การเชื่อมต่ออุปกรณ์ ST-Link/V2 กับบอร์ด STM32F411CEU ที่ขา SWD

STM32 (SWD) | ST-Link/V2
GND <-----> GND
SCK <-----> SWCLK
DIO <-----> SWDIO
3V3 <-----> 3.3V

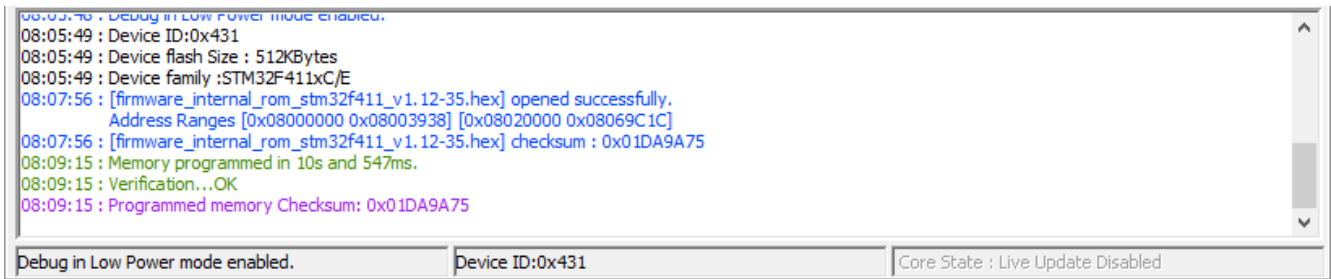
การเชื่อมต่อดังวิธี **SWD** จะใช้สายไฟเชื่อมต่อ 4 เส้น ระหว่างบอร์ด **STM32** กับอุปกรณ์ **ST-Link V2**

เมื่อเชื่อมต่อกับบอร์ด **STM32** ผ่านทาง **ST-Link V2 USB** กับคอมพิวเตอร์ได้แล้ว ให้เปิดโปรแกรม **STM32 ST-Link Utility** จากนั้นไปที่เมนู **Target > Connect** ถ้าสามารถเชื่อมต่อได้ จะปรากฏข้อความระบุ **Device** ที่ตรวจพบ



รูปภาพ: ST-Link Utility และเปิดไฟล์ .hex สำหรับ WeAct STM32F411CEU

ถัดไปให้เปิดไฟล์ .hex โดยทำคำสั่งจากเมนู **File > Open File** แล้วเลือกไฟล์ .hex ที่ต้องการจะโปรแกรมไปยังบอร์ดไมโครคอนโทรลเลอร์ จากนั้นทำขั้นตอน **Target > Program & Verify** (หรืออาจทำขั้นตอน **Erase Chip** ก่อนก็ได้ เพื่อเคลียร์หน่วยความจำ **Flash** ทั้งหมด)



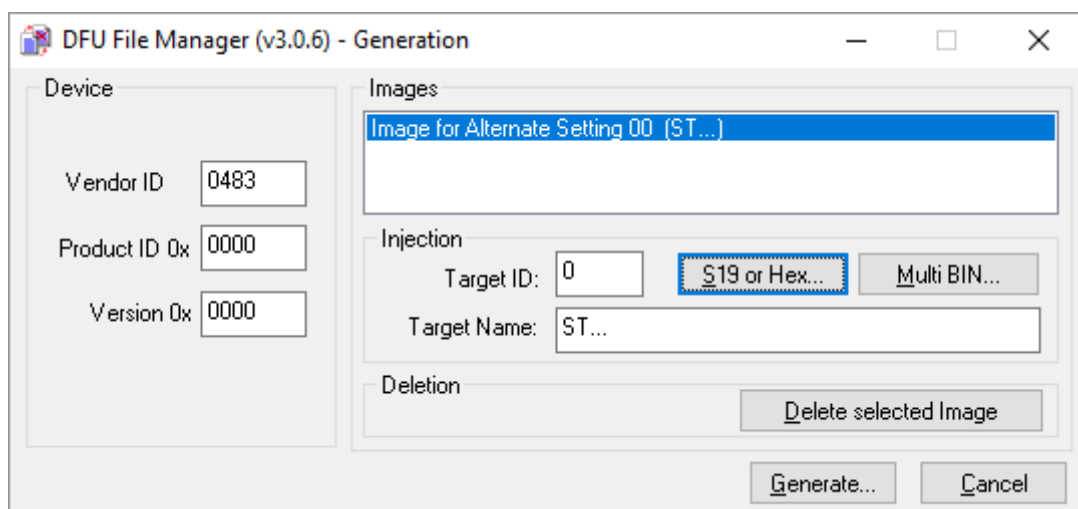
รูปภาพ: ข้อความแสดงสถานะการทำงานของ ST-Link Utility

การโปรแกรมด้วยวิธี DFU สำหรับ Windows

เมื่อได้ติดตั้งโปรแกรมชื่อ **DfuSe** ไว้สำหรับการอัปโหลดไฟล์เฟิร์มแวร์ด้วยวิธี DFU (USB device firmware upgrade) ถัดไปให้เตรียมอุปกรณ์ฮาร์ดแวร์ดังนี้ (ในกรณีที่ใช้บอร์ด **WeAct STM32F411CEU6**)

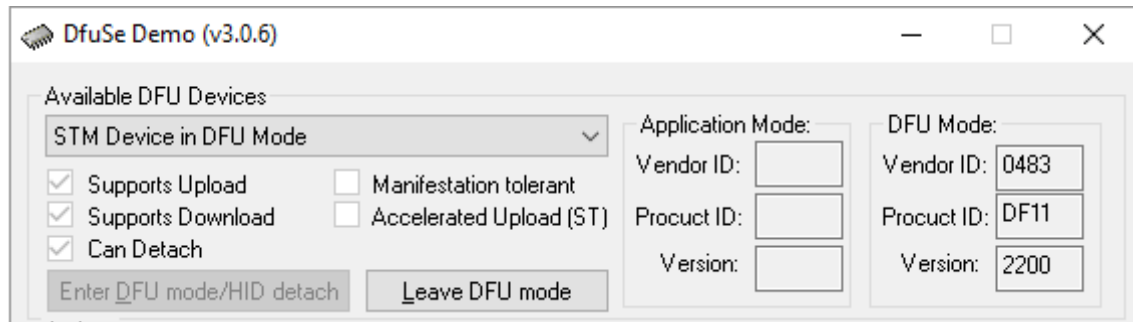
- เชื่อมต่อขา **A10 (PA0/USB_FS_ID)** ด้วยสาย **Jumper Wire** กับตัวต้านทาน **10k** แบบ **Pullup** ไปยัง **3.3V**
- เสียบสาย **USB-C** เชื่อมต่อบอร์ดไมโครคอนโทรลเลอร์กับคอมพิวเตอร์
- กดปุ่ม **BOOT0** กดค้างไว้ กดปุ่ม **RESET** แล้วจึงปล่อยปุ่ม **RESET** และ **BOOT0** ตามลำดับ

ถ้าต้องการจะแปลงไฟล์ **.hex** เป็น **.dfu** ก็ให้ใช้โปรแกรมชื่อ **DFU File Manager** ของ **DfuSe** โดยกดปุ่ม **"S19 or Hex"** แล้วเลือกไฟล์ **.hex** ตามด้วยการกดปุ่ม **Generate**



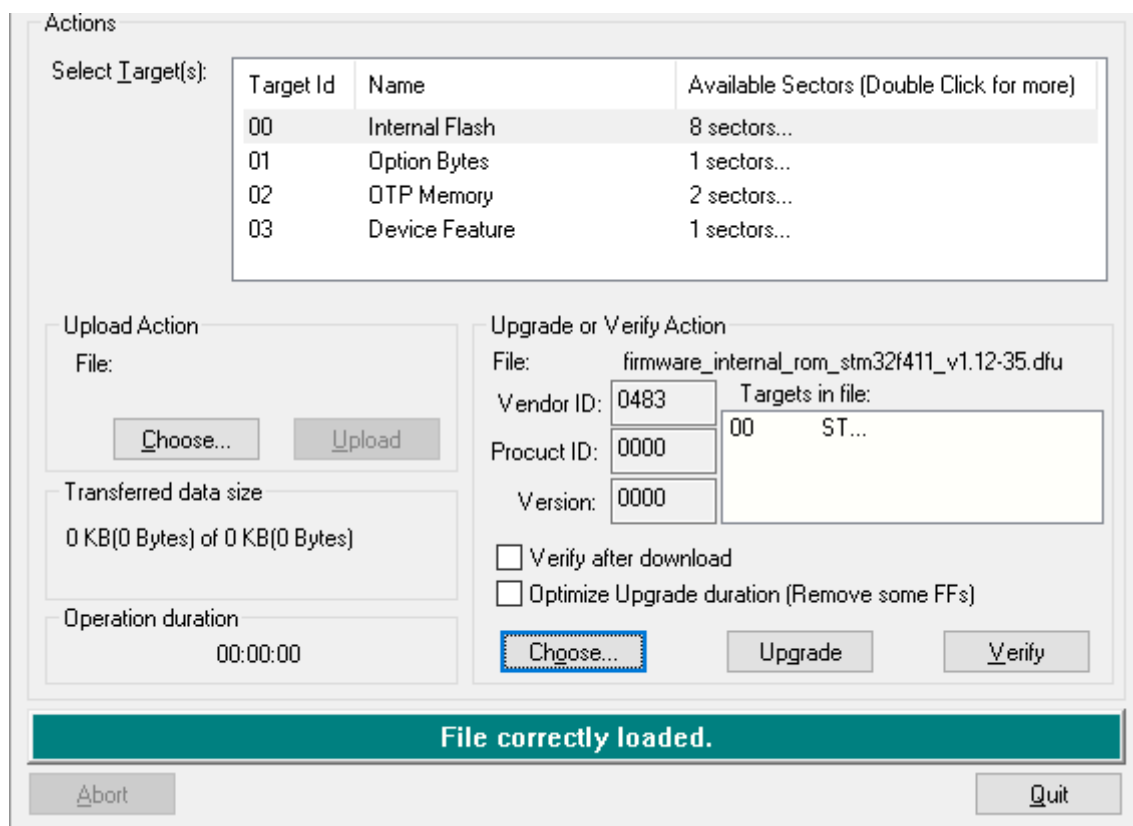
รูปภาพ: การแปลงไฟล์ .hex ให้เป็น .dfu

ถัดไปให้เปิดโปรแกรม **DfuSe Demo** ถ้าเชื่อมต่อบอร์ด **STM32** แล้วอยู่ในโหมด **DFU** จะมองเห็น **Vendor ID: 0483, Product ID: DF11** และ **Version: 2200**

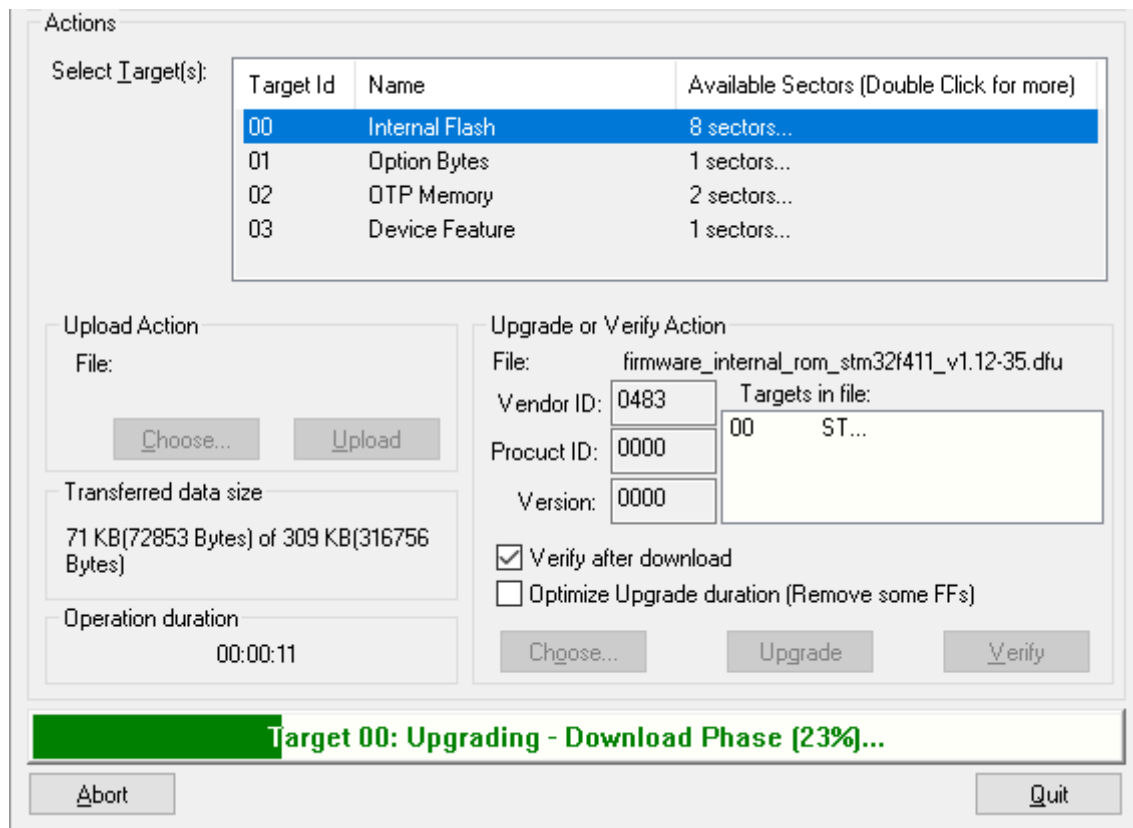


รูปภาพ: แสดงสถานะการมองเห็นอุปกรณ์ STM32 Dfu Mode

กดปุ่ม **Choose...** ในส่วนของ **Upgrade or Verify Action** เลือกไฟล์ .dfu แล้วกดปุ่ม **Upgrade** เพื่อทำขั้นตอนสุดท้าย



รูปภาพ: แสดงสถานะการเปิดไฟล์ .dfu ได้สำเร็จ

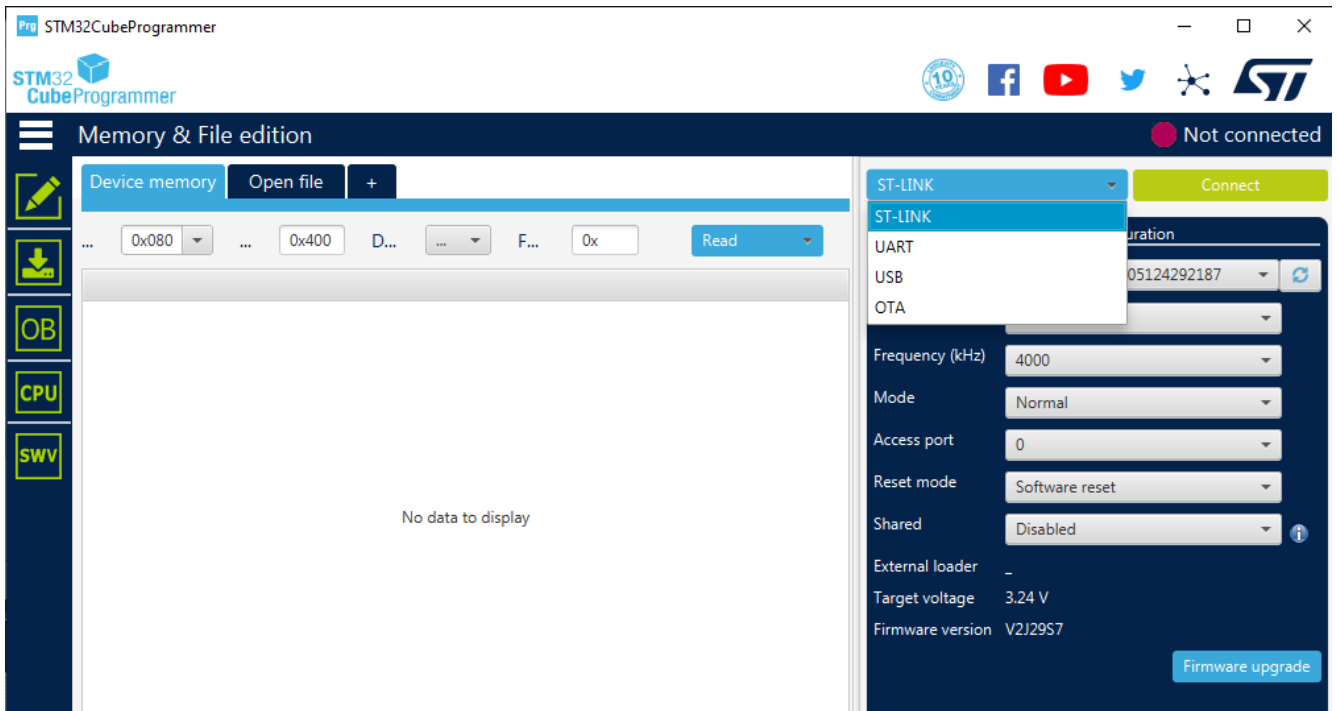


รูปภาพ: ขั้นตอนการเขียนไฟล์ .dfu ไปยังบอร์ดไมโครคอนโทรลเลอร์

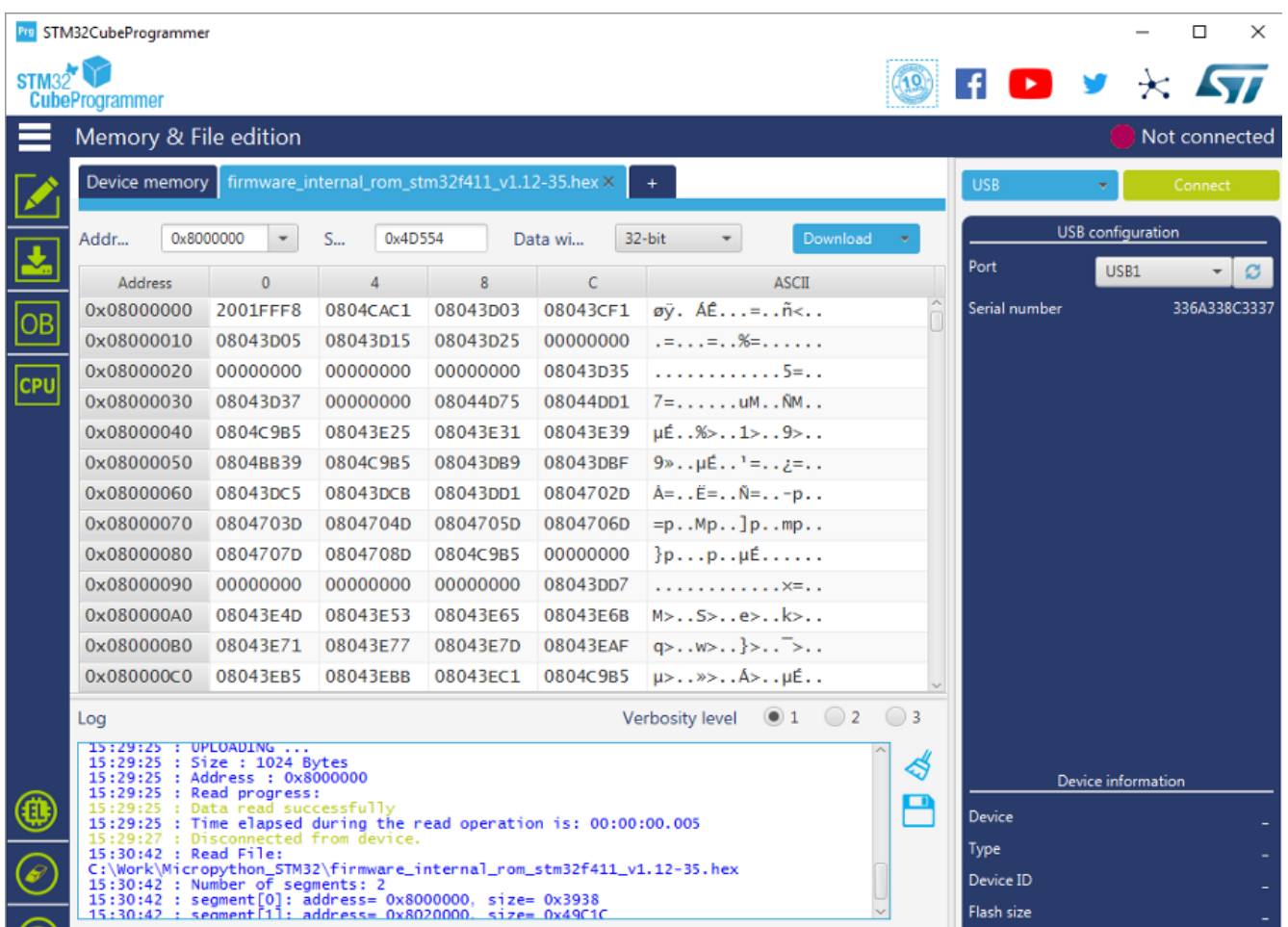
การใช้งาน STM32CubeProgrammer สำหรับ Windows

เราสามารถใช้ซอฟต์แวร์ **STM32CubeProgrammer** ซึ่งรองรับการใช้งานได้ทั้ง **SWD** (ใช้ร่วมกับ **ST-Link V2**) และ **DFU (USB)** และเลือกใช้ไฟล์ **.hex** ได้เลย (ไม่ต้องแปลงเป็น **.dfu**)

การใช้งานโปรแกรมนี้ ก็ทำได้ง่าย เช่น เลือกใช้ **ST-Link** แล้วก็เชื่อมต่อกับบอร์ดไมโครคอนโทรลเลอร์ (**Connect**) จากนั้นเลือกไฟล์ **.hex** (**Open File**) แล้วก็กดปุ่ม **Download** เพื่อเขียนข้อมูลไปยังอุปกรณ์เป้าหมาย

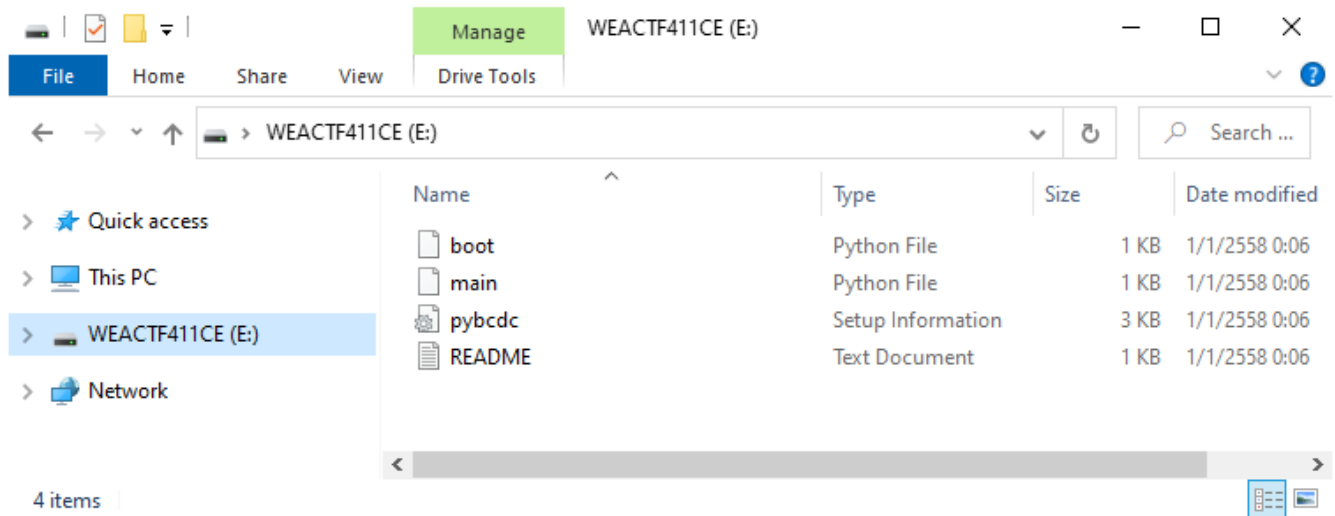


รูปภาพ: การเลือกอุปกรณ์ ST-LINK



รูปภาพ: การแฟลชไฟล์ .hex / .dfu ด้วยโปรแกรม STM32CubeProgrammer

เมื่อติดตั้งเฟิร์มแวร์สำหรับไมโครไพธอนได้สำเร็จแล้ว และนำบอร์ด **STM32F411CEU6** ไปเชื่อมต่อเข้ากับพอร์ต **USB** ของคอมพิวเตอร์ (ใช้สาย **USB Type-C**) จะมองเห็นไดรฟ์ชื่อ **STM32F411CE**



รูปภาพ: รายการไฟล์ภายในไดรฟ์ของบอร์ด STM32F411CE-MicroPython

การใช้โปรแกรม stlink-tools สำหรับ Linux

ตัวอย่างการทำการคำสั่งต่อไปนี้ ได้ทดลองใช้กับ **Raspbian OS** สำหรับบอร์ด **Raspberry Pi 4** แต่ก็นำไปใช้กับคอมพิวเตอร์ที่ทำงานด้วย **Linux Ubuntu** ได้เหมือนกัน

เริ่มต้นด้วยการคำสั่งติดตั้งโปรแกรมนี้

```
$ sudo apt-get install stlink-tools
```

จากนั้นให้เชื่อมต่ออุปกรณ์ **ST-Link V2** กับบอร์ด **STM32F411CEU6** และพอร์ต **USB** ของคอมพิวเตอร์ แล้วทำการคำสั่งต่อไปนี้ เพื่อลบข้อมูลทั้งหมดในหน่วยความจำ **Flash (Full Chip Erase)** และเขียนไฟล์ **.hex** ไปยังอุปกรณ์ ตามลำดับ

```
$ st-flash erase
```

```
$ st-flash --reset write firmware_internal_rom_stm32f411_v1.12-540.hex 0x80000000
```



```
pi@rpi4: ~  
pi@rpi4:~ $ st-info --probe  
Found 1 stlink programmers  
serial: 553f6d06483f50512429213f  
openocd: "\x55\x3f\x6d\x06\x48\x3f\x50\x51\x24\x29\x21\x3f"  
flash: 524288 (pagesize: 16384)  
sram: 131072  
chipid: 0x0431  
descr: F4 device (low power) - stm32f411re  
pi@rpi4:~ $ st-flash erase  
st-flash 1.5.1  
2020-10-03T09:48:41 INFO common.c: Loading device parameters....  
2020-10-03T09:48:41 INFO common.c: Device connected is: F4 device (low power) - stm32f411re, id 0x10006431  
2020-10-03T09:48:41 INFO common.c: SRAM size: 0x20000 bytes (128 KiB), Flash: 0x80000 bytes (512 KiB) in pages of 16384 bytes  
Mass erasing.....  
pi@rpi4:~ $
```

รูปภาพ: ตัวอย่างการทำการคำสั่ง st-flash erase

```
pi@rpi4: ~  
pi@rpi4:~ $ st-flash --reset write firmware_internal_rom_stm32f411_v1.12-540.hex 0x8000000  
st-flash 1.5.1  
2020-10-03T09:50:09 INFO common.c: Loading device parameters....  
2020-10-03T09:50:09 INFO common.c: Device connected is: F4 device (low power) - stm32f411re, id 0x10006431  
2020-10-03T09:50:09 INFO common.c: SRAM size: 0x20000 bytes (128 KiB), Flash: 0x80000 bytes (512 KiB) in pages of 16384 bytes  
2020-10-03T09:50:09 INFO common.c: Attempting to write 438696 (0x6b1a8) bytes to stm32 address: 134217728 (0x8000000)  
Flash page at addr: 0x08060000 erasedEraseFlash - Sector:0x7 Size:0x20000  
2020-10-03T09:50:18 INFO common.c: Finished erasing 8 pages of 131072 (0x20000) bytes  
2020-10-03T09:50:18 INFO common.c: Starting Flash write for F2/F4/L4  
2020-10-03T09:50:18 INFO flash_loader.c: Successfully loaded flash loader in sram  
enabling 32-bit flash writes  
size: 32768  
size: 32768  
size: 32768  
size: 32768  
size: 32768  
size: 32768  
size: 32768  
size: 32768  
size: 32768
```

รูปภาพ: ตัวอย่างการทำการคำสั่ง st-flash write

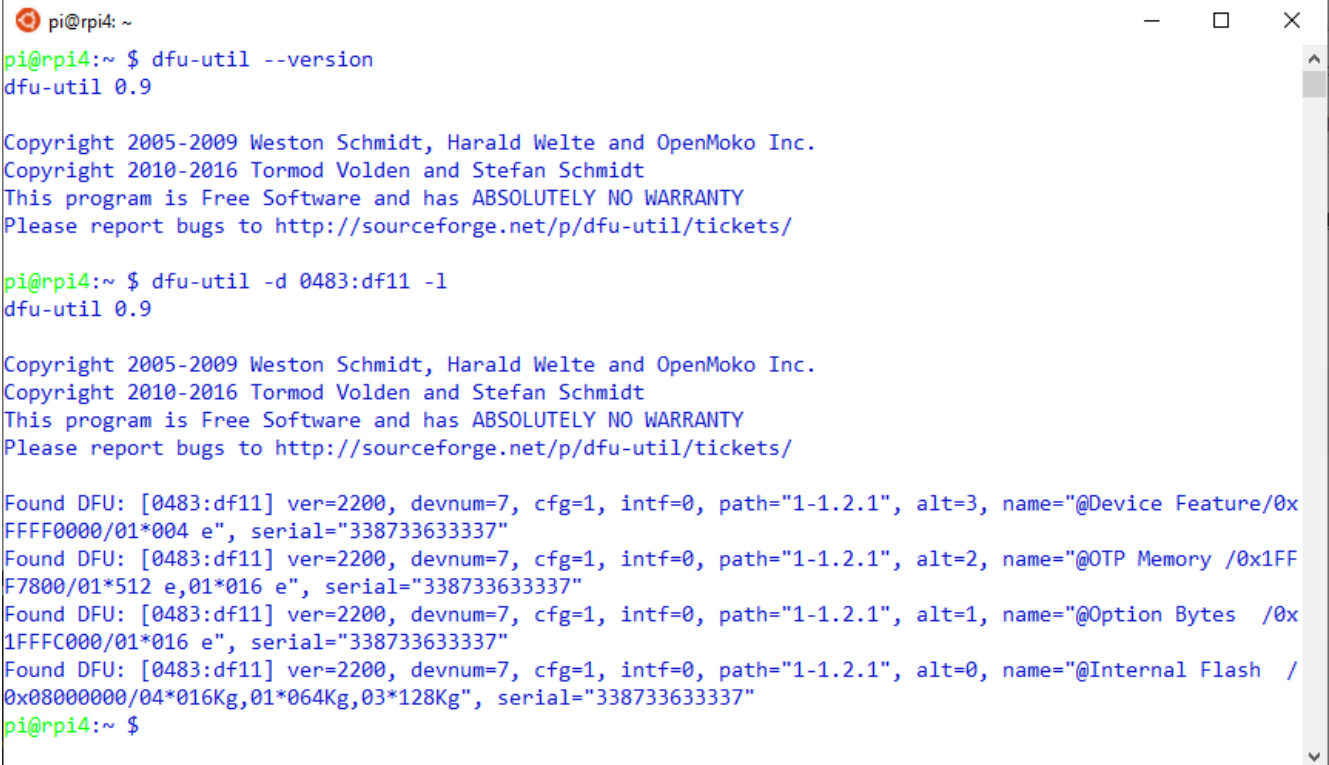
การใช้โปรแกรม dfu-util สำหรับ Linux

ตัวอย่างการทำการคำสั่งต่อไปนี้ ได้ทดลองใช้กับ Raspbian OS สำหรับบอร์ด Raspberry Pi 4 แต่ก็สามารถนำไปใช้กับคอมพิวเตอร์ที่ทำงานด้วย Linux Ubuntu ได้เหมือนกัน

เริ่มต้นด้วยการทำให้บอร์ด **STM32** เข้าโหมด **DFU Bootloader Mode** ก่อน จากนั้นจึงทำตามคำสั่งดังต่อไปนี้ เพื่อติดตั้งโปรแกรม แล้วจึงตรวจสอบดูว่า สามารถมองเห็นอุปกรณ์ในโหมด **DFU** หรือไม่

```
$ sudo apt-get install dfu-util
```

```
$ dfu-util -d 0483:df11 -l
```



```
pi@rpi4: ~  
pi@rpi4:~ $ dfu-util --version  
dfu-util 0.9  
  
Copyright 2005-2009 Weston Schmidt, Harald Welte and OpenMoko Inc.  
Copyright 2010-2016 Tormod Volden and Stefan Schmidt  
This program is Free Software and has ABSOLUTELY NO WARRANTY  
Please report bugs to http://sourceforge.net/p/dfu-util/tickets/  
  
pi@rpi4:~ $ dfu-util -d 0483:df11 -l  
dfu-util 0.9  
  
Copyright 2005-2009 Weston Schmidt, Harald Welte and OpenMoko Inc.  
Copyright 2010-2016 Tormod Volden and Stefan Schmidt  
This program is Free Software and has ABSOLUTELY NO WARRANTY  
Please report bugs to http://sourceforge.net/p/dfu-util/tickets/  
  
Found DFU: [0483:df11] ver=2200, devnum=7, cfg=1, intf=0, path="1-1.2.1", alt=3, name="@Device Feature/0x  
FFFF0000/01*004 e", serial="338733633337"  
Found DFU: [0483:df11] ver=2200, devnum=7, cfg=1, intf=0, path="1-1.2.1", alt=2, name="@OTP Memory /0x1FF  
F7800/01*512 e,01*016 e", serial="338733633337"  
Found DFU: [0483:df11] ver=2200, devnum=7, cfg=1, intf=0, path="1-1.2.1", alt=1, name="@Option Bytes /0x  
1FFFC000/01*016 e", serial="338733633337"  
Found DFU: [0483:df11] ver=2200, devnum=7, cfg=1, intf=0, path="1-1.2.1", alt=0, name="@Internal Flash /  
0x08000000/04*016Kg,01*064Kg,03*128Kg", serial="338733633337"  
pi@rpi4:~ $
```

รูปภาพ: ข้อความแสดงสถานะการเชื่อมต่อกับอุปกรณ์ในโหมด DFU

จากนั้นให้ทำตามคำสั่งเพื่อเขียนไฟล์ **.dfu** ไปยังหน่วยความจำ **Flash** ภายในชิป **STM32** (ในตัวอย่างนี้ได้ดาวน์โหลดไฟล์ **.dfu** ที่เป็นเฟิร์มแวร์ของไมโครคอนโทรลเลอร์รุ่น **v1.12** สำหรับบอร์ด **WeAct STM32F411CEU6** มายังคอมพิวเตอร์ของผู้ใช้แล้ว)

```
$ dfu-util -a 0 -D ./firmware_internal_rom_stm32f411_v1.12-540.dfu
```

```
pi@rpi4: ~  
Match vendor ID from file: 0483  
Match product ID from file: df11  
Opening DFU capable USB device...  
ID 0483:df11  
Run-time device DFU version 011a  
Claiming USB DFU Interface...  
Setting Alternate Setting #0 ...  
Determining device status: state = dfuERROR, status = 10  
dfuERROR, clearing status  
Determining device status: state = dfuIDLE, status = 0  
dfuIDLE, continuing  
DFU mode device DFU version 011a  
Device returned transfer size 2048  
DfuSe interface name: "Internal Flash "  
file contains 1 DFU images  
parsing DFU image 1  
image for alternate setting 0, (2 elements, total size = 322424)  
parsing element 1, address = 0x08000000, size = 14784  
Download [=====] 100% 14784 bytes  
Download done.  
parsing element 2, address = 0x08020000, size = 307624  
Download [=====] 100% 307624 bytes  
Download done.  
done parsing DfuSe file  
pi@rpi4:~ $
```

รูปภาพ: ข้อความที่แสดงสถานะการทำงานของ dfu-util

การใช้โปรแกรม OpenOCD สำหรับ Linux

โปรแกรม **openocd** รองรับการใช้งานอุปกรณ์ **ST-LINK V2** เราจะมาลองใช้งานเป็นตัวอย่าง ตัวอย่างการทำคำสั่งต่อไปนี้ ได้ทดลองใช้กับ **Raspbian OS** สำหรับบอร์ด **Raspberry Pi 4** แต่ก็สามารถนำไปใช้กับคอมพิวเตอร์ที่ทำงานด้วย **Linux Ubuntu** ได้เหมือนกัน

เริ่มต้นด้วยการติดตั้งใช้งานโปรแกรมนี้ก่อน

```
$ sudo apt-get install openocd libftdi-dev
```

ถัดไปให้สร้างไฟล์ **./target_stm32f4.cfg** ซึ่งเป็นการตั้งค่าการใช้งาน (**OpenOCD Configuration Settings**) ดังนี้

```
source [find interface/stlink.cfg]
transport select hla_swd
source [find target/stm32f4x.cfg]
telnet_port disabled
gdb_port disabled
adapter srst delay 100
reset_config none separate
adapter speed 980
```

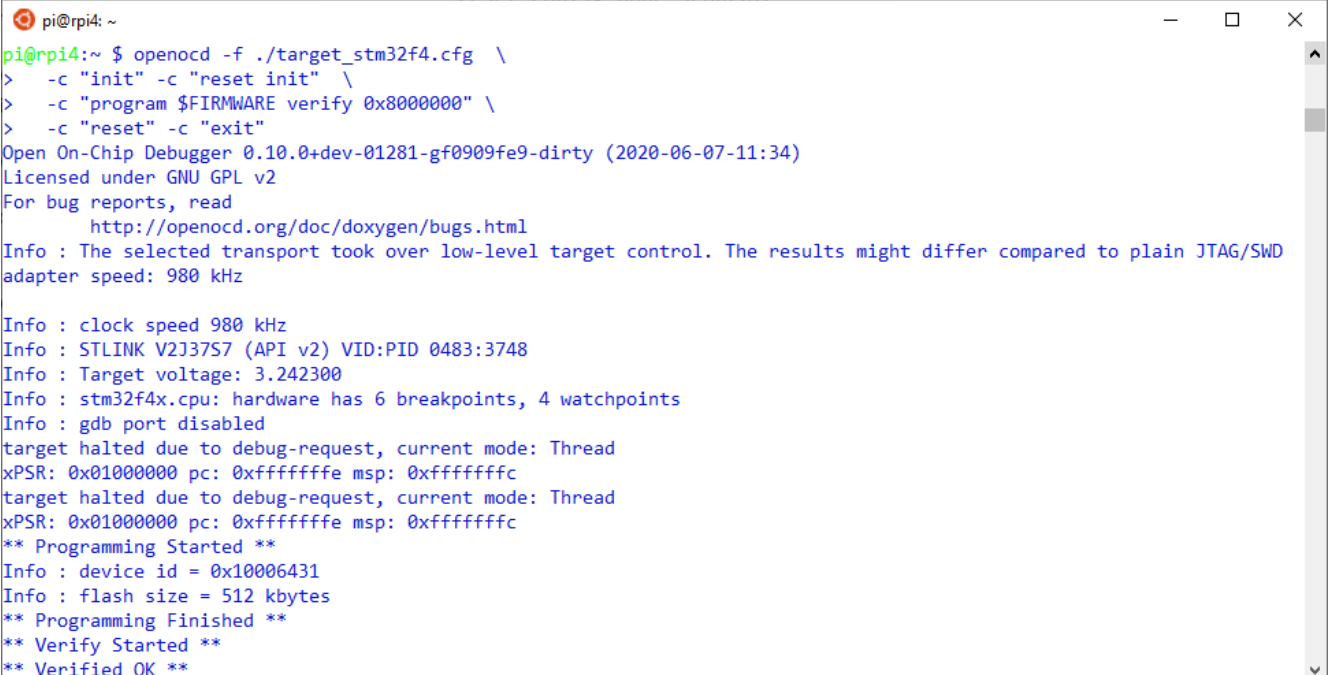
เชื่อมต่ออุปกรณ์ให้พร้อม จากนั้นทำคำสั่งเพื่อลบข้อมูลในหน่วยความจำ **Flash** ของอุปกรณ์เป้าหมาย โดยทำคำสั่งดังนี้

```
$ openocd -f ./target_stm32f4.cfg \
-c "init" -c "reset init" \
-c "stm32f2x unlock 0; reset halt" \
-c "flash erase_sector 0 0 last" \
-c "reset" -c "shutdown"
```

คำสั่งถัดไปคือ การเขียนข้อมูลจากไฟล์เฟิร์มแวร์ไปยังอุปกรณ์เป้าหมาย

```
$ export FIRMWARE=./firmware_internal_rom_stm32f411_v1.12-540.hex
```

```
$ openocd -f ./target_stm32f4.cfg \
-c "init" -c "reset init" \
-c "program $FIRMWARE verify 0x8000000" \
-c "reset" -c "exit"
```



```
pi@pi4: ~
pi@pi4:~ $ openocd -f ./target_stm32f4.cfg \
> -c "init" -c "reset init" \
> -c "program $FIRMWARE verify 0x8000000" \
> -c "reset" -c "exit"
Open On-Chip Debugger 0.10.0+dev-01281-gf0909fe9-dirty (2020-06-07-11:34)
Licensed under GNU GPL v2
For bug reports, read
  http://openocd.org/doc/doxygen/bugs.html
Info : The selected transport took over low-level target control. The results might differ compared to plain JTAG/SWD
adapter speed: 980 kHz

Info : clock speed 980 kHz
Info : STLINK V2J3757 (API v2) VID:PID 0483:3748
Info : Target voltage: 3.242300
Info : stm32f4x.cpu: hardware has 6 breakpoints, 4 watchpoints
Info : gdb port disabled
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0xffffffff msp: 0xffffffff
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0xffffffff msp: 0xffffffff
** Programming Started **
Info : device id = 0x10006431
Info : flash size = 512 kbytes
** Programming Finished **
** Verify Started **
** Verified OK **
```

รูปภาพ: แสดงข้อความเมื่อทำคำสั่ง openocd เพื่อเขียนไฟล์ .hex

โดยสรุป

เราได้เห็นวิธีการและเครื่องมือประเภทซอฟต์แวร์ต่าง ๆ ที่เป็นตัวเลือกสำหรับการติดตั้งเฟิร์มแวร์ของไมโครไพธอน เพื่อให้ใช้ไต่ทำงานกับบอร์ดไมโครคอนโทรลเลอร์ **STM32** เช่น **STM32F411CEU6**