Unbricking POC

These instructions are for unbricking the POC NED, POC Form Factor and POC-AC.

Equipment

You'll need:

- POC device (included)
- POC charging cable (included)
- POC EDL cable with the red button (included)
- USB-C female to female connector **(buy separately)**
  - Option A
    - https://www.amazon.com/Cellularize-Extension-Thunderbolt-Compatible-Connector/dp/B07S292X2V
  - Option B
    - https://www.amazon.com/gp/product/B079LYHNSR/
    - https://www.amazon.com/gp/product/B07FT9MDX4/

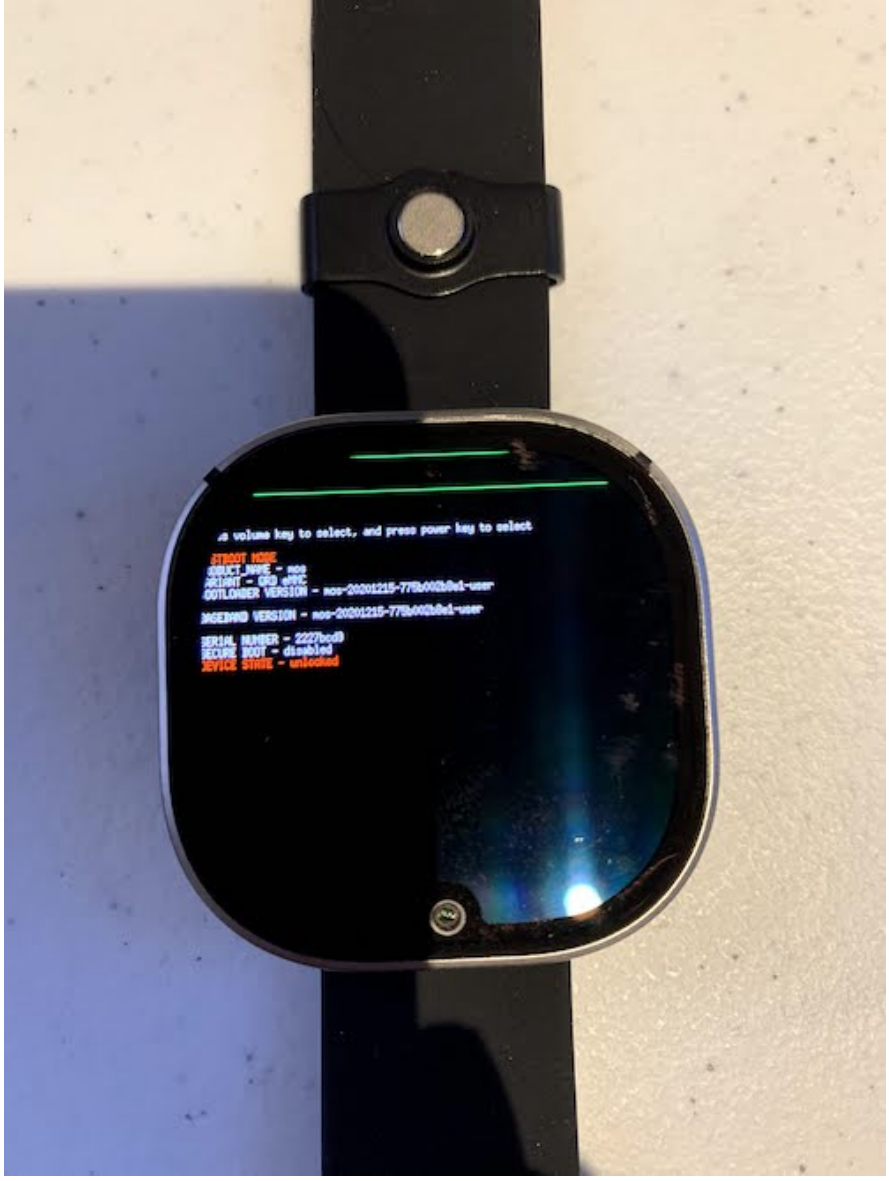
Note: Ironically there is atleast n=2 data point where this entire process worked with just the regular charging cable (and had no success with the EDL cable)! So definitely try with that too.

Is my device bricked?

- If your battery is low, try charging it.
- Hold the power and camera buttons for ~10 seconds.
  - **NOTE: for POC NED users, you need a pen to press the power button between the display and the dev board. Make sure you hear a click using a pen. And make sure that you hold for 10 seconds when you hear it. https://fb.workplace.com/groups/144766930169063/permalink/321804052465349/ may be a good option if you don't have steady hands.**

FASTBOOT MODE

If you see the following screen, then you should be able to run `maui flash` and carry on with your day.



In n=1 case, just running `maui flash` didn't properly unbrick. In this case, follow steps 1-5 here and run `./flash_all.py -p -w` note that this may require a serial number, which can be done by adding `-s <device serial number>` and the serial number found by running `fastboot devices` while in fastboot mode

QUALCOMM USB DIAGNOSTICS MODE

If the screen won't turn on, try the following command:

```
ioreg -p IOUSB
```

```
$ ioreg -p IOUSB
+-o Root  <class IORegistryEntry, id 0x100000100, retain 18>
  +-o AppleUSBEHCI Root Hub Simulation@1d000000  <class AppleUSBRootHubDevice, id 0x100000572, registered, matched, active, busy 0 (69 ms), retain 12>
  | +-o IOUSBHostDevice@1d100000  <class AppleUSBDevice, id 0x100000574, registered, matched, active, busy 0 (0 ms), retain 13>
  |   +-o BRCM20702 Hub@1d110000  <class AppleUSBDevice, id 0x100000596, registered, matched, active, busy 0 (0 ms), retain 12>
  |     +-o Bluetooth USB Host Controller@1d113000  <class AppleUSBDevice, id 0x1000005c5, registered, matched, active, busy 0 (0 ms), retain 17>
  +-o AppleUSBXHCI Root Hub Simulation@00000000  <class AppleUSBRootHubDevice, id 0x100000584, registered, matched, active, busy 0 (41 ms), retain 10>
    +-o Savant Elite2 Foot Pedal@00100000  <class AppleUSBDevice, id 0x100003a33, registered, matched, active, busy 0 (1 ms), retain 16>
    +-o USB2.0 Hub             @00200000  <class AppleUSBDevice, id 0x10013dea3, registered, matched, active, busy 0 (0 ms), retain 13>
      +-o QHSUSB__BULK@00210000  <class AppleUSBDevice, id 0x10014c759, registered, matched, active, busy 0 (0 ms), retain 12>
```

If you see something like `QHSUSB__BULK...` , that means the OS is broken and you need to enter EDL (Emergency DownLoad mode). Follow the EDL instructions below.

EDL (Emergency DownLoad Mode)

EDL CABLE WITH RED BUTTON

You'll need to enter the special Qualcomm EDL mode with a red push button connector.

- You should have two of them (USB-A-to-C and USB-C-to-C).
- People have had more luck with the USB-A-to-C one.

The red button is a latch and can be in two states: connected and disconnected. Pushing and releasing the button toggles the state. If the cap feels lose, the button is in the **pressed/disconnected** state, which has the USB lines shorted, if the user presses again, the button should be unlatched (the red cap should now feel tight) and this un-shorts the USB lines and returns the cable to normal, this is the **unpressed/connected** state.

- The red button is in the **connected** state if the device is powered on and connected to the computer and shows up in `ioreg -p IOUSB` as `QHSUSB__BULK...` or `SDM429W-QRD`.
- `SDM429W-QRD` signifies that the device is in EDL mode. (On OSX, ioreg seems to never switch to `SDM429W-QRD` so looking up the device id in lsusb seems the way to go: it should switch to `05c6:9008` from `05c6:900e` when the button did its job.)
- The red button is in the **disconnected** state if the device is powered on and connected to the computer but doesn't show up in `ioreg -p IOUSB`.

REGULAR CHARGING CABLE

We have at least an n=2 data point report of just trying the above steps with the regular charging cable (EDL cable didnt help) but had to retry *lots* of times to get it in the mode. In that case just ignore the **connected+ disconnected** part of the flashing instructions below.

I was able to get into the mode by discharging the device all the way and then reconnecting it.

POC NED P1 DEVICE

This has anecdotally worked for n=2 people, feel free to reach out to grasberger@ with questions.

To get a POC NED P1 device into EDL mode:

1. Hold down the "Forced USB Boot" Button
2. Plug in the UART cable
3. Plug in the USB-C cable
4. Wait 5 seconds (exact timing unknown)
5. Release the "Forced USB Boot" Button
6. The device should show up when you run `ioreg -p IOUSB`
7. Follow the instructions below for flashing a QFIL image

`lsusb`

```
Bus 020 Device 001: ID 1050:0407 1050 YubiKey OTP+FIDO+CCID
Bus 020 Device 051: ID 05c6:9008 Qualcomm, Inc QHSUSB__BULK
Bus 000 Device 002: ID 0bda:0412 Realtek Semiconductor Corp. 2-Port USB 3.1 Hub
```

`ioreg -p IOUSB`

```
  +-o AppleUSBXHCI Root Hub Simulation@14000000  <class AppleUSBRootHubDevice, id 0x100000672, registered, matched, active, busy 0 (85 ms), retain 10>
  | +-o YubiKey OTP+FIDO+CCID@14400000  <class AppleUSBDevice, id 0x100000674, registered, matched, active, busy 0 (0 ms), retain 17>
  | +-o QHSUSB__BULK@14100000  <class AppleUSBDevice, id 0x10044a39e, registered, matched, active, busy 0 (2 ms), retain 12>
  +-o AppleUSBXHCI Root Hub Simulation@02000000  <class AppleUSBRootHubDevice, id 0x100001ea0, registered, matched, active, busy 0 (49 ms), retain 8>
```

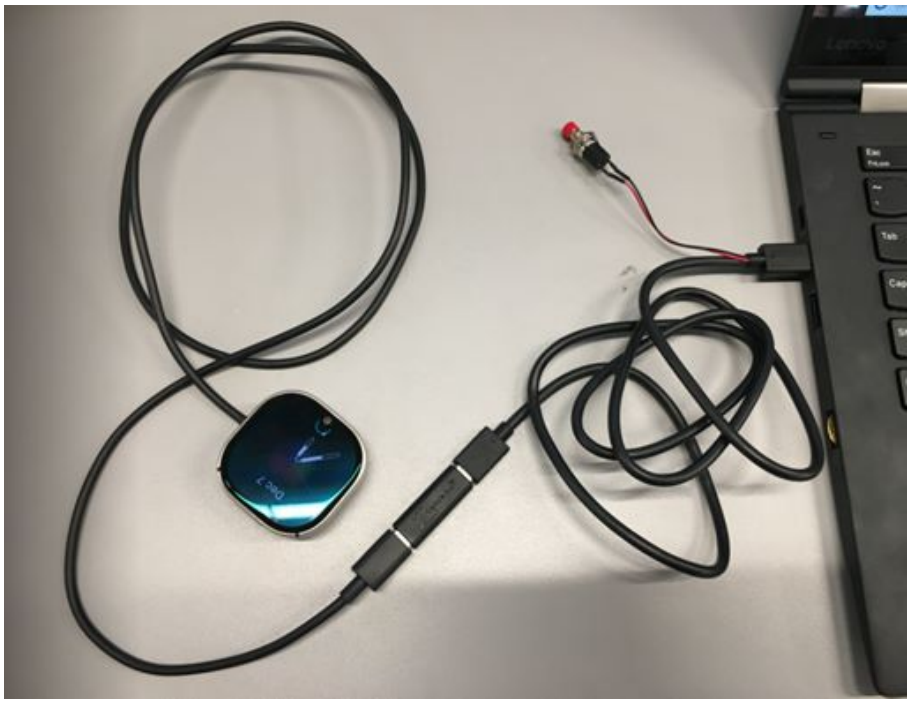
INSTRUCTIONS

If you're confused or stuck, please check out one of these resources:
https://fb.workplace.com/groups/2520879348005870/permalink/3819750231452102

See a video demonstration here: https://fb.workplace.com/100009406885935/videos/2887268591596675/

On OSX, install lsusb with `brew install lsusb` then run `while :; do lsusb 2> /dev/null | grep 'Qual';done` to get roughly the same output as on the video demo, but look for ID changing from `05c6:900e` to `05c6:9008`

1. Make sure the red button is in the **disconnected** state (see above for instructions).
2. Connect Milan on the dock, hold both buttons for 10s to force a HW reset just in case if the device was not previously off. (Anecdotally, holding just the power (circular) button on the FF POC could work even better.)
3. Press the button to unlatch and return the cable to **connected** state, `ioreg -p IOUSB should return SDM429W-QRD` (OSX users: our marker is device ID == `05c6:9008`)
4. Download the latest QFIL image from here: https://fburl.com/oculus/6dubb8t5
    a. Unzip the folder and enter the directory
    b. Run `python flash_qfil_package.py`
5. Then the device should reboot into a functional state
6. If it doesn't, try a different qfil package



FAQ

QFIL FAILS WITH AN ERROR SIMILAR TO "{ERROR: COULD NOT WRITE TO 'USB:', WINDOWS API WRITEFILE FAILED! YOUR DEVICE IS PROBABLY *NOT* ON THIS PORT, ATTEMPTED 100 TIMES}

Milan Emergency Download Mode (EDL) gets into a weird state and it doesn't accept any data from the QFIL flashing script sometimes. To fix, hold power button on Milan to turn it off, and turn it on again to reset EDL (will take a few attempts). Once you do this the QFIL script works again.

You can monitor whether you are back in EDL by using "watch -n1 ioreg -p IOUSB" and waiting for QHSUSB to appear. If you don't have the "watch" command line program installed, you can use "brew install watch" to install it.

References

- Original discussion: https://fb.workplace.com/groups/546044966017058/permalink/697666340854919/
- Follow-up discussion: https://fb.workplace.com/groups/3848061321935258/permalink/5523602211047819

« Live Network Access

Milan EDBv2 »

)}