Firmware Flashing

- Flashing MeshCore Firmware
- Flashing Meshtastic Firmware
- Flashing Troubleshooting

Flashing MeshCore Firmware

Flashing MeshCore Firmware

MeshCore firmware can be installed via the web flasher (easiest), with command-line flashing tools (esptool for ESP32, UF2 drag-and-drop for nRF52), or OTA (over-the-air) for updates on already-running devices. All methods are covered below.

Method 1: Web Flasher (Recommended)

The web flasher at **flasher.meshcore.io** (the canonical flasher, run by the MeshCore core team and linked from the official docs - URL accurate as of 2026-06-13) requires a Chromium-based browser (Chrome or Edge). Firefox does not support the WebSerial API and will not work. *Note: flasher.meshcore.co.uk is a separate, downstream flasher associated with the MeshOS variant - for standard MeshCore, use flasher.meshcore.io.*

1. Open **flasher.meshcore.io** in Chrome or Edge.
2. Connect your device via USB. Use a **data-capable** USB cable. Charge-only cables (common with power banks) will not expose the serial port. If the device does not appear, try a different cable first.
3. Hold the BOOT button while plugging in (ESP32 devices) or double-tap the reset button (nRF52 devices) to enter bootloader mode.
4. Select your device type from the dropdown.
5. Select the firmware variant:
   - **Companion** - pairs with a phone app over BLE or USB
   - **Repeater** - autonomous relay node, no interaction needed
   - **Room Server** - store-and-forward message hub
6. Click **Flash**. The process takes 1 - 2 minutes. Do not disconnect during flashing.
7. After flashing completes, configure the device via the MeshCore app or CLI.

Method 2: Command-Line Flashing

Note: `meshcore-cli` does **not** flash firmware - it connects to an already-running node to configure and interact with it (see Post-Flash Configuration below). Command-line *flashing* uses the standard chip-flashing tools instead, which is useful for bulk deployments or when the web flasher is unavailable.

ESP32 boards (V3, T-Beam, Station G2, etc.) - esptool: Install Python 3 first, then install esptool. On many systems use `pip3` if `pip` is not found; on recent Linux/macOS a bare system `pip install` is blocked (PEP 668), so use a virtual environment or `pipx`:

```
pip3 install esptool   # or: pipx install esptool
```

Put the board in bootloader mode (hold BOOT while connecting USB), then write the MeshCore merged firmware .bin. Replace the port and filename to match your system:

```
esptool.py --port COM3 write_flash 0x0 meshcore-firmware.bin       # Windows
esptool.py --port /dev/ttyUSB0 write_flash 0x0 meshcore-firmware.bin  # Linux/macOS
```

nRF52 boards (RAK4631, Heltec T114, T-Echo, etc.) - UF2 drag-and-drop: Double-tap the reset button to enter the UF2 bootloader. The board appears as a USB mass-storage drive; copy the MeshCore `.uf2` file onto that drive. The board flashes and reboots automatically. No command-line tool is required.

Method 3: OTA Update (ESP32 devices)

For devices already running MeshCore firmware, OTA updates avoid needing a USB connection. ESP32 boards update over a temporary Wi-Fi access point. See **Flashing MeshCore Firmware OTA: The Definitive Guide** in the MeshCore book for the full walkthrough.

1. In the MeshCore app, open the Command Line for your device.
2. Type: `start ota`
3. The device will create a Wi-Fi hotspot named **MeshCore-OTA** (it may appear as "MeshCore OTA").
4. Connect your phone or computer to that Wi-Fi network.
5. Open a browser and navigate to **http://192.168.4.1/update**
6. Upload the **non-merged** firmware file (.bin) - not the merged .bin, which is for first-time USB flashing. Wait for the device to reboot.

Method 4: OTA Update (nRF52 devices)

nRF52-based devices use the Nordic DFU protocol for OTA updates. Installing the OTAFIX bootloader (from flasher.meshcore.io) first makes BLE OTA far more reliable. See **Flashing MeshCore Firmware OTA: The Definitive Guide** for the full walkthrough.

1. Place the device in OTA DFU mode (with OTAFIX installed, hold the button while resetting; otherwise issue `start ota` in the MeshCore app Command Line).
2. Use the **nRF Device Firmware Update** app (available for Android/iOS; it appears as "DFU" in your app list).
3. Select the firmware .zip DFU package and transfer. The standard DFU flow handles transfer automatically; leave the app's transfer/packet-receipt settings at their defaults

unless the MeshCore release notes for your board specify otherwise.

Bootloader Entry by Device

Device	Method
Most ESP32 devices (V3, V4, T-Beam)	Hold BOOT button while connecting USB
nRF52 (T-Echo, RAK WisBlock, Wio series)	Double-tap reset button quickly
LilyGo T-Deck variants	Depress trackball while connecting USB
Heltec V3	Uses a CP2102 USB-serial bridge - install the Silicon Labs CP210x driver first
Heltec V4	Native USB CDC (the CP2102 was removed on V4) - no external USB-serial driver needed

Post-Flash Configuration (meshcore-cli)

After flashing, the device needs basic configuration before it will function on the network. You can configure it from the MeshCore app, or from a terminal with `meshcore-cli`, which connects to a *running* node (it does not flash firmware). Install it with `pip3 install meshcore-cli` (use a venv or `pipx` on PEP 668 systems), then connect using one of:

```
meshcore-cli -s COM3            # serial (Windows); use -s /dev/ttyUSB0 on Linux/macOS
meshcore-cli -a <BLE address>   # BLE by address (or -S to scan and pick from a selector)
meshcore-cli -t 192.168.1.100 -p 5000   # TCP/IP (default port 5000)
```

Then set the basics:

- Set device name (used to identify you in the mesh)
- Set region/frequency. For North America select the US region: this constrains the radio to the 902-928 MHz band (commonly shorthanded "915 MHz") and is the primary compliance control - the US region preset enforces the band edges and caps TX power to the FCC Part 15 limit. Do not select a non-US region or disable the cap.
- Set TX power. For unlicensed US operation, conducted output must not exceed 30 dBm (1 W) and must be reduced further for antennas above 6 dBi (so total EIRP stays within the 36 dBm / 47 CFR 15.247 ceiling). The US region preset normally caps power correctly, but verify on high-power hardware (e.g. Station G2 or 2 W modules), which can default or be set above the legal limit. Reduce power for indoor testing.
- For Repeater variant: set the repeater name and ensure auto-start is enabled

Flashing Meshtastic Firmware

Flashing Meshtastic Firmware

Meshtastic firmware is flashed via the web flasher at **flasher.meshtastic.org** or via the Meshtastic Python CLI. The process is similar to MeshCore but has some differences in device selection and channels.

Web Flasher

1. Open **flasher.meshtastic.org** in Chrome or Edge. Firefox will not work (no WebSerial support).
2. Connect the device via a USB data cable. Enter bootloader mode:
   - ESP32: hold BOOT button while plugging in
   - nRF52: double-tap reset button
   - T-Deck: depress trackball while connecting
3. Select your device from the dropdown. If your device is not listed, check the Meshtastic hardware support page.
4. Choose firmware channel:
   - **Stable** - recommended for most users; well-tested
   - **Alpha** - latest features, may have bugs
5. Click Flash. The process takes 1 - 3 minutes depending on device.

Driver Requirements

Some devices require USB-serial drivers before the OS will recognise them. If unsure, check the silkscreen label on the USB-serial chip near the USB connector:

Chip	Driver	Common Devices
CP2102 / CP210x	Silicon Labs CP210x driver	Heltec V3, some T-Beam variants
CH340/CH341	CH340 driver (Windows/macOS)	Some LilyGo boards (and occasional clone units of other boards)

Chip	Driver	Common Devices
Native USB CDC	No driver needed	Heltec V4 (the CP2102 was removed on V4)
nRF52840 UF2 bootloader	No driver needed - appears as a USB mass-storage drive	T-Echo, RAK WisBlock, most nRF52

Note: the Heltec WiFi LoRa 32 **V3** uses a CP2102 bridge (install the Silicon Labs CP210x driver); the **V4** uses native USB CDC and needs no external driver. Occasional units or clones may ship with a CH340 - if the CP210x driver does not produce a COM port, check the chip silkscreen.

First-Time Configuration

After flashing Meshtastic, use the Meshtastic app (Android/iOS) or web client to configure:

- **Region:** Set to US (902-928 MHz, commonly shorthanded "915 MHz") for North America. Beyond interoperability (a wrong region means you cannot communicate with local nodes), the region setting is the legal control: the US region enforces the FCC band edges (902-928 MHz) and caps TX power to the legal Part 15 maximum (30 dBm conducted, reduced for antennas above 6 dBi). Selecting a non-US region to gain power or change frequency can put you outside US law.
- **Role:** CLIENT for normal/personal devices; ROUTER only for genuine fixed infrastructure (the Meshtastic project discourages casual use of ROUTER, as a misused router role can degrade the mesh). The older ROUTER_CLIENT role was retired in firmware 2.3.15 - use CLIENT (or CLIENT_MUTE) instead, including for mobile/vehicle nodes. See the Meshtastic role documentation for the current guidance.
- **Long name / short name:** How you appear to other users on the mesh.
- **Channel:** Must match other nodes you want to communicate with. Default channel works for public networks.

Re-Flashing Between Firmware Versions

You can move between Meshtastic stable and alpha, or between Meshtastic and MeshCore, at any time. Re-flashing normally does not physically damage the hardware, but a wrong or interrupted flash - especially using the wrong bootloader/.uf2 on an nRF52 board - can leave the device temporarily unbootable; see Flashing Troubleshooting for recovery. Configuration is reset when flashing a new firmware type, so note your settings before switching.

Flashing Troubleshooting

Flashing Troubleshooting

Most flashing failures fall into a small set of categories. Work through this table before assuming the device is damaged.

Troubleshooting Table

Symptom	Likely Cause	Fix
Device not detected by browser or OS	Charge-only USB cable; wrong USB port; missing drivers	Try a different cable (data-capable); try a different USB port; install CH340 or CP2102 drivers; try a different computer
Device detected but flash fails immediately	Not in bootloader mode	Hold BOOT while connecting (ESP32) or double-tap reset (nRF52); consult device-specific instructions
Device won't boot after flashing	Wrong firmware build selected	Verify you selected the correct device in the flasher; re-flash with correct build
Flashed wrong variant (e.g., Repeater instead of Companion)	User error	Re-flash with correct variant; no permanent damage
ESP32 completely unresponsive / "bricked"	Corrupted flash	Hold BOOT button → connect USB → run: `esptool.py --port COM3 erase_flash` (substitute your actual port, e.g. `/dev/ttyUSB0` on Linux or `/dev/cu.usbserial-*` on macOS) → re-flash firmware. On newer installs the command may be `esptool` or `python -m esptool`; run `pip install esptool` first.
nRF52 unresponsive	Corrupted firmware	Double-tap reset button to enter DFU mode → reflash via DFU
Linux: "Permission denied" on /dev/ttyUSB0	User not in dialout group	`sudo usermod -a -G dialout $USER` then log out and back in
Linux: Web flasher cannot connect	udev rules / ACL issue	`setfacl -m u:$USER:rw /dev/ttyUSB0`

Symptom	Likely Cause	Fix
macOS: Device not appearing in /dev/	Missing USB-serial driver (depends on the board's USB chip)	Identify the chip first. CH340/CH9102 boards: install `CH34xVCPDriver` from wch.cn (or the Homebrew formula). CP210x boards (many Heltec/older devices): install the Silicon Labs CP210x VCP driver. Newer boards with native USB (nRF52 DFU, ESP32-S3 native CDC, e.g. Heltec V4) need no driver and enumerate as `/dev/cu.usbmodem*`. Recent macOS bundles some drivers, but a missing-device symptom usually means the matching VCP driver is not installed.
Flash completes but device shows wrong region or settings	Old config preserved in flash	Perform a factory reset via the app or by flashing with "erase before flash" option checked

esptool.py Emergency Erase

If an ESP32 device is completely unresponsive and normal bootloader entry fails:

```
pip install esptool
esptool.py --port COM3 erase_flash # Windows
esptool.py --port /dev/ttyUSB0 erase_flash # Linux
```

After erasing, the chip will be blank. Re-flash the firmware normally via the web flasher. (On newer esptool installs the command may be invoked as `esptool` or `python -m esptool` instead of `esptool.py`.)

Identifying Your Serial Port

Windows: Device Manager → Ports (COM & LPT) → look for CH340 or CP210x device. Note the COM number (e.g., COM5).

Linux: Run `ls /dev/tty*` before and after plugging in the device. The new entry is your device (typically /dev/ttyUSB0 or /dev/ttyACM0).

macOS: Run `ls /dev/cu.*`. Look for cu.usbserial-* or cu.wchusbserial*.