Setting Up Zephyr RTOS Development Environment and Building/Flashing the Blinky Sample for Nordic nRF52840 Dongle

January 2026

1 Overview

This guide describes the process of setting up a Zephyr RTOS development environment on a Raspberry Pi (Debian Bookworm), building MCUboot as the primary bootloader for USB DFU support, building the basic "blinky" LED sample for the Nordic Semiconductor nRF52840 Dongle (bare configuration), signing the image, and flashing/programming via SWD using OpenOCD with Raspberry Pi GPIO pins or updating via MCUboot USB serial recovery. This example requires no special equipment, no hardware edits, all hardware available online, and no soldering,

The nRF52840 Dongle is a compact USB dongle featuring the nRF52840 SoC, commonly used for Bluetooth Low Energy and other wireless applications in Zephyr projects. Using MCUboot replaces the native Nordic bootloader and enables secure firmware updates over USB.

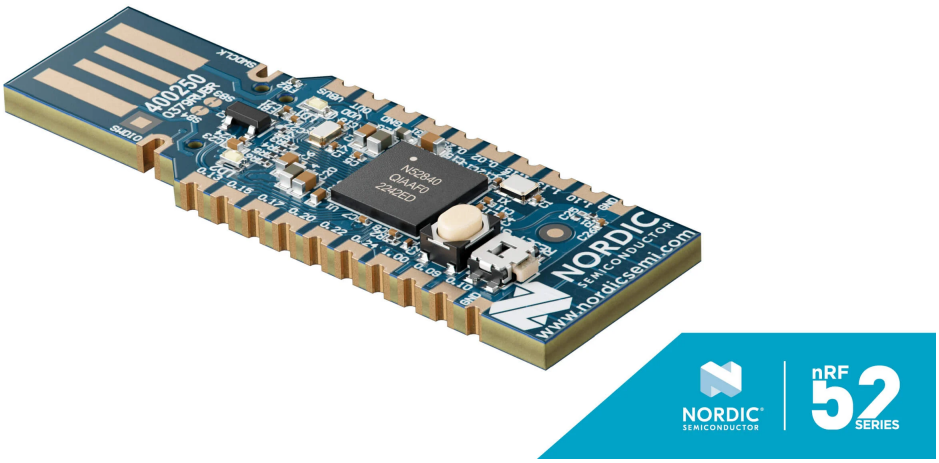

Figure 1: Official Nordic nRF52840 Dongle

2 Hardware Connections for SWD Flashing

To flash the dongle using OpenOCD on Raspberry Pi GPIO, connect the following pins:

- Raspberry Pi GPIO 24 (SWDIO) → Dongle SWDIO
- Raspberry Pi GPIO 25 (SWCLK) → Dongle SWCLK

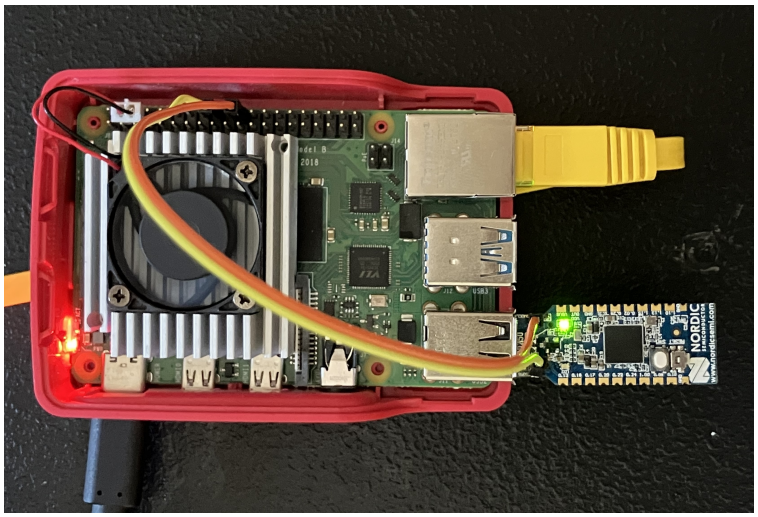

Figure 2: nRF52840 Dongle (Zephyr Documentation)

- Raspberry Pi GND → Dongle GND
- (Optional) Raspberry Pi 3.3V → Dongle VCC if not powered via USB

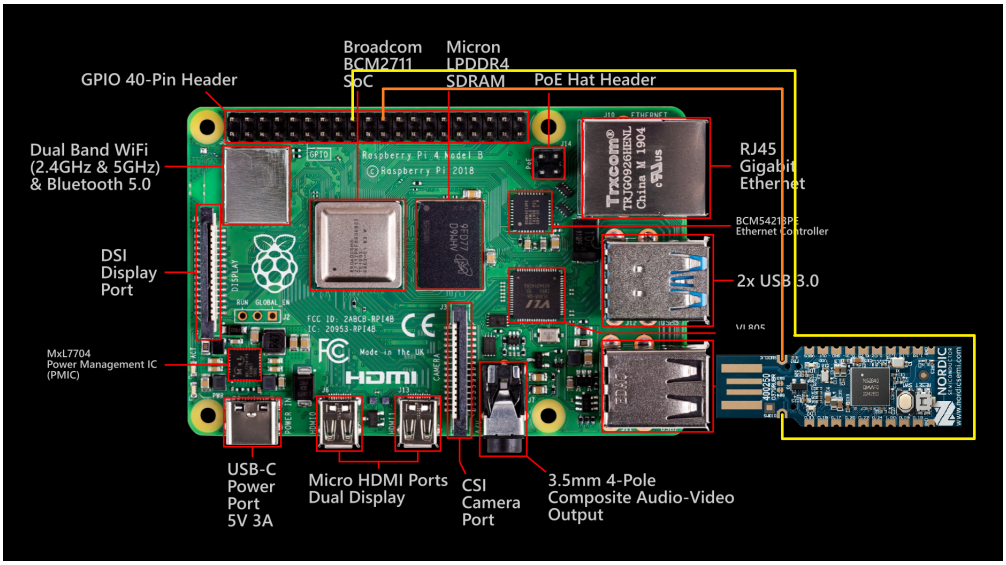

Figure 3: Example SWD wiring between Raspberry Pi and nRF52840 Dongle

3 Running the Setup Script

The script `setup-zephyr-dev-1.0.0.sh` automates the installation of required tools and the Zephyr workspace.

```
chmod +x setup-zephyr-dev-1.0.0.sh
sudo ./setup-zephyr-dev-1.0.0.sh
```

Key actions performed:

- Installs system packages (`git`, `cmake`, `ninja-build`, `openocd`, etc.)

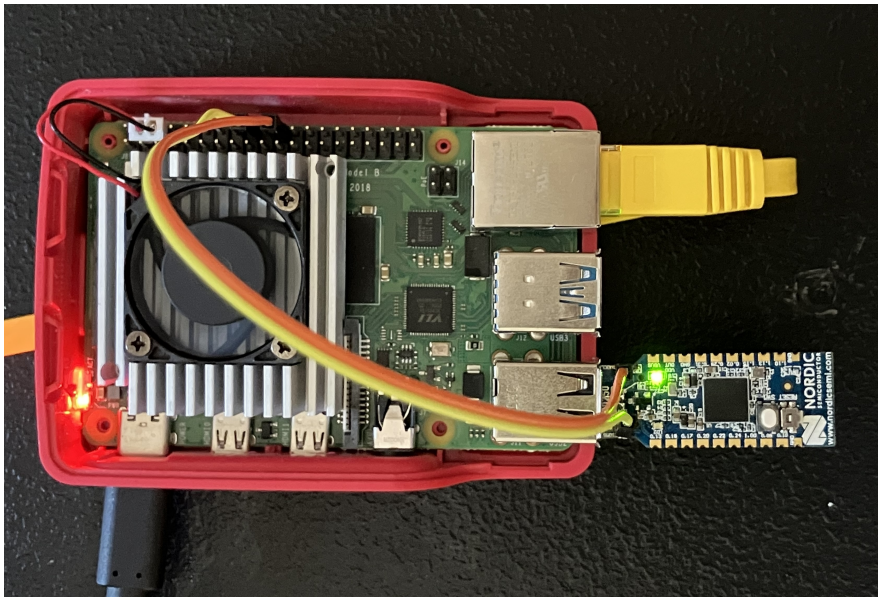

Figure 4: nRF52840 Dongle Pinout Diagram

- Sets up Zephyr workspace using `west`
- Downloads and installs Zephyr SDK 0.17.4
- Installs Go and the `mcumgr` CLI
- Generates OpenOCD configuration for GPIO SWD
- Configures MCUboot (optional for DFU)

After completion, reboot the system:

```
sudo reboot
```

4 Building and Flashing MCUboot

Log in as the `dev` user after reboot. The virtual environment activates automatically.

Create or edit `mcuboot`$_{serial.conf}$ `in bootloader/mcuboot/boot/zephyr/` *with the following content for USB serial recover*

```
CONFIG_MCUBOOT_SERIAL=y
CONFIG_BOOT_SERIAL_CDC_ACM=y
CONFIG_USB_DEVICE_STACK=y
CONFIG_USB_DEVICE_PID=0x1001
CONFIG_USB_DEVICE_PRODUCT="MCUboot"
CONFIG_MCUBOOT_INDICATION_LED=y
CONFIG_WARN_DEPRECATED=n  # Suppress USB deprecation warnings
CONFIG_MCUBOOT_LOG_LEVEL_OFF=y  # Disable logging to fix choice warnings
```

Build MCUboot for the bare nRF52840 Dongle:

```
cd ~/zephyrproject
west build -p always -b nrf52840dongle/nrf52840/bare \
  -d build-mcuboot bootloader/mcuboot/boot/zephyr \
  DEXTRA_CONF_FILE=mcuboot_serial.conf
```

If Kconfig warnings cause abortion, set:

```
export ZEPHYR_KCONFIG_WARN_AS_ERROR=0
```

and retry the build.

Flash MCUboot via SWD (replaces native bootloader):

```
sudo openocd -f ~/openocd.cfg \
  -c "init; reset halt; nrf5 mass_erase; \
      program build-mcuboot/zephyr/zephyr.hex verify; \
      reset; exit"
```

Successful output confirms device detection, mass erase, programming, verification, and reset. After flashing, unplug and replug the dongle while holding the button (SW1) to enter recovery mode|the LED should fade slowly, and it enumerates as /dev/ttyACM0.

5 Building the Blinky Sample

Build the blinky sample with MCUboot support:

```
west build -p always -b nrf52840dongle/nrf52840/bare \
  -d build/blinky zephyr/samples/basic/blinky \
  -- -DCONFIG_BOOTLOADER_MCUBOOT=y
```

Successful build output includes memory usage and generates build/blinky/zephyr/zephyr.bin.

Figure 5: Example of Blinky LED behavior on a Zephyr-compatible board

6 Signing and Uploading the Image

Sign the blinky binary using MCUboot's default key:

```
west sign -d build/blinky -t imgtool -- --key bootloader/mcuboot/root-rsa-2048.pem
```

Enter MCUboot recovery mode: Plug in the dongle while holding the button (SW1). It should appear as /dev/ttyACM0.

Upload, verify, confirm, and reset using mcumgr:

```
mcumgr --conntype=serial --connstring='dev=/dev/ttyACM0,baud=115200' \
  image upload -e build/blinky/zephyr/zephyr.signed.bin
mcumgr --conntype=serial --connstring='dev=/dev/ttyACM0,baud=115200' \
  image list
mcumgr --conntype=serial --connstring='dev=/dev/ttyACM0,baud=115200' \
  image confirm
mcumgr --conntype=serial --connstring='dev=/dev/ttyACM0,baud=115200' \
  reset
```

After reset, the onboard LED should blink periodically.

7 Conclusion

This setup provides a complete Zephyr development environment on Raspberry Pi for the nRF52840 Dongle with MCUboot as the primary bootloader. MCUboot enables secure USB DFU updates via mcumgr, replacing the native Nordic bootloader. The blinky sample serves as a ‘‘hello world’’ verification. For production, generate custom signing keys and consider additional security features.