Cortex-M Bare-Metal Training: STM32 and libopencm3

Cortex-M microcontrollers programmed without vendor HALs or IDEs, down to what runs before main().

4 days	Beginner to intermediate	On site or remote	English or French
Duration	Level	Format	Language

Overview

This course teaches bare-metal programming on Arm Cortex-M microcontrollers with an open source toolchain and the libopencm3 library, without vendor HALs or IDEs. It goes from clocks and GPIOs to the communication peripherals, then down to linker scripts, flash handling, the Cortex-M architecture and a custom bootloader.

What You Will Learn

- Configure clocks, GPIOs, interrupts and timers with libopencm3
- Drive UART, I2C, SPI and CAN, with interrupts and DMA
- Write linker scripts and place code and data in memory
- Program the flash and protect it with option bytes
- Write a bootloader for a Cortex-M microcontroller
- Reduce power consumption with the low-power modes and the RTC

Who should attend

Firmware engineers who want full control over their microcontroller code, and engineers moving to Cortex-M from other architectures.

Prerequisites

Good knowledge of C. No prior Cortex-M experience is needed.

Program

The course starts with the peripherals and ends with what happens before main().

Day 1

- Microcontrollers, libopencm3 and the STM32 family
- Reset and clock control
- GPIOs, pin control and multiplexing
- Interrupts and the NVIC, the SysTick timer

Day 2

- Timers and their operating modes
- Peripheral DMA
- U(S)ART handling with DMA

Day 3

- I2C and SPI, in polling, interrupt and DMA modes
- CAN bus

Day 4

- Memory layout and linker scripts
- Flash memory handling and option bytes
- The Cortex-M architecture and exception handling, assembly
- Writing a bootloader
- Power management and the RTC

Labs and Hardware

Labs run on STM32 Nucleo boards. Sessions take place on site, in France, Belgium, Switzerland, Germany and beyond, with the hardware provided by LABCSMART, or remotely. Remote attendees work on our build server, each with their own SSH account and workspace, and run the labs on QEMU machines and on real boards. The exercises are drawn from our own client projects. Courses are given in English or French.

Who Teaches This Course

The course is designed and given by the founder of LABCSMART, an embedded Linux and Linux kernel engineer with more than eleven years of industrial experience. He contributes upstream to the Linux kernel for Renesas RZ SoCs, in the PCIe, audio, Ethernet, thermal, DMA and clock subsystems, and wrote two books published by Packt, *Linux Device Driver Development, Second Edition* (2022) and *Mastering Linux Device Driver Development* (2021), presented on our publications page.

Plan a session for your team. Every course is made of independent modules. A team can drop the modules it does not need and add others in their place, and we also build courses on demand around a given SoC, board or project. Contact: contact@labcsmart.com, https://labcsmart.com/company/contact/. Course page: https://labcsmart.com/training/stm32-bare-metal-libopencm3/