STR14 - STM32G0

This course describe the STM32G0 architecture and practical examples

Objectives

- Understand the STM32G0 family, Cortex-M0+ basics, and the device memory map.
- Bring up a board cleanly (CubeMX/CubeIDE), configure clocks/reset, and verify timing.
- Use GPIO/EXTI, timers (PWM, capture, encoder), and watchdogs confidently.
- Stream data efficiently with DMA/DMAMUX and timer-triggered transfers.
- Acquire and process analog data with ADC (oversampling, watchdog) and COMP.
- Implement robust comms on USART/LPUART, I²C, and SPI (IRQ/DMA patterns).
- Design and measure low-power behavior (Sleep/Stop/Standby with RTC/LPTIM).
- (When supported) Bring up UCPD (USB-C/PD), USB FS device, and CAN-FD on G0B1/C1.
- Apply CRC checks and reset-cause diagnostics for production robustness.

Course Environment

- Theoretical course
  - PDF course material (in English) supplemented by a printed version for face-to-face courses.
  - Online courses are dispensed using the Teams video-conferencing system.
  - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- Practical activities
  - Practical activities represent from 40% to 50% of course duration.
  - Code examples, exercises and solutions
  - For remote trainings:
    - One Online Linux PC per trainee for the practical activities.
    - The trainer has access to trainees' Online PCs for technical and pedagogical assistance.
    - QEMU Emulated board or physical board connected to the online PC (depending on the course).
    - Some Labs may be completed between sessions and are checked by the trainer on the next session.
  - For face-to-face trainings:
    - One PC (Linux ou Windows) for the practical activities with, if appropriate, a target board.
    - One PC for two trainees when there are more than 6 trainees.
  - For onsite trainings:
    - An installation and test manual is provided to allow preinstallation of the needed software.
    - The trainer come with target boards if needed during the practical activities (and bring them back at the end of the course).
- Downloadable preconfigured virtual machine for post-course practical activities
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Course Outline

Day 1

ARM Cortex-M0+ overview (core concepts)

- Programmer’s mode
- Exception model
- Faults on ARMv6-M
- Instruction set at a glance
- Debug

Exercise: Preemption
Exercise: Exception Management

STM32G0 SoC architecture

- Family lines: G0x0 / G0x1 / G0B1 / G0C1; package options and pin-mux/AF principles.
- Bus/memory map: Flash, SRAM, peripheral regions, PPB; where to put const/config.
- Device identifiers: UID, Flash size registers; why we log them in production.
- Boot straps & Option Bytes overview
- Basic electrical notes

Project bring-up (CubeMX/CubeIDE), startup & linker

- HAL vs LL usage strategy (size/latency vs convenience).
- Startup & vector table overview; where to place user init.
- Linker script essentials: sections, memory regions, placing a const config block.
- Assert & Error paths: routing assert_failed() / Error_Handler() to UART safely.
- Minimal logging: lightweight printf vs ITM vs ring-buffer UART.

Exercise: Clean template
Exercise: Config section
Exercise: Error/Assert path

GPIO, EXTI & SYSCFG

- Output modes
- EXTI mapping
- Input protection & safe states on power-up
- SYSCFG basics that affect I/O/interrupt routing.

Exercise: GPIO / EXTI configuration

RCC, Reset & Clock Control

- Clock sources
- PLL path
- SYSCLK mux & prescalers
- Peripheral kernel clocks (CCIPR)
- MCO output for measurement; CSS and clock interrupts (CIER/CIFR/CICR).

Exercise: Output MCO; verify frequencies with a logic analyzer/DMM.

Timers & LPTIM (+ Watchdogs)

- Timer blocks
  - Advanced (complementary/BDTR)

    - General-purpose
    - basic
- PWM modes
- Trigger/clock routing between timers (ETR/ITRx) and DMA requests
- LPTIM for ultra-low-power periodic wake
- RTC
- IWDG (independent) and WWDG (windowed)

Exercise: Timer example

Day 2

DMA / DMAMUX

- DMA basics
    - Directions
    - Circular/normal
    - Events
- DMAMUX request mapping; request generators/sync (if present on device).
- Throughput vs latency tuning
- Error handling
- Typical pipelines

Exercise: DMA modes and request mapping

ADC & Analog (COMP, DAC* by variant)

- ADC basics
    - Resolution
    - Sampling times
    - Sequences
    - Modes
- Oversampling to increase ENOB
- External triggers from timers; DMA coupling and latency notes.
- Analog watchdog(s) for threshold events
- VREFINT / TEMP sensor; (variant) VBAT; COMP outputs to timers/EXTI; (variant) DAC

Exercise: ADC and DMA

USART/LPUART (Serial)

- Modes
- Oversampling 16/8; impact on baud accuracy for HSI vs HSE clocking.
- Blocking vs IRQ vs DMA; RX ring buffer with idle-line detection.
- Stop-mode wake-up via RX; when to switch kernel clock via CCIPR.
- Framing/overrun errors; robust recovery without lockups.

Exercise: UART DMA practical example

SPI

- Master/slave basics; polarity/phase (CPOL/CPHA) and timing windows.
- Data size options; NSS hardware vs software; multi-slave selection patterns.
- Full-duplex vs simplex; DMA transfers and double-buffering.
- Throughput vs CPU load; using a timer/EXTI to pace transactions.
- Debugging MOSI/MISO with a logic analyzer; common mis-wires.

I²C

- Sm/Fm/Fm+ modes up to 1 Mbit/s; clock config & rise-time considerations.
- Addressing (7/10-bit); repeated start; memory-like transactions.

- Glitch filtering (analog/digital) and its trade-offs.
- Timeouts & bus-clear to recover from stuck SCL/SDA; clock stretching.
- Using Stop-mode with I²C wake; when it’s safe.

CRC & Basic Security Aids

- Hardware CRC unit: polynomial, reflect options (device-specific).
- Typical uses: image CRC at boot, packet integrity in comms stacks.
- Placement of CRC fields in Flash; speed vs code-size.
- Integrating CRC with DMA for long buffers.
- Relationship with RDP/PCROP: integrity vs confidentiality.

Exercise: Compute CRC over the application region at startup

(Optional) FDCAN on G0B1/G0C1

- CAN-FD vs classic CAN; nominal/data phases and bit-rate switching.
- Message RAM layout; dedicated RAM sizing and filters.
- Acceptance filters: mask/list ranges, priority implications.
- Loopback/silent modes for bring-up; error counters.
- Transceiver & termination requirements; EMC notes.

Day 3

Low-Power Modes & Measurement

- Sleep vs Stop vs Standby: retention, wake sources, startup latencies.
- GPIO states to minimize leakage; analog/pull configuration when asleep.
- Peripheral clock gating and CCIPR choices that aid low-power.
- Measuring current correctly (DMM shunt vs power analyzer); sampling pitfalls.
- Policy: when to enter/exit low-power; watchdog coordination.

Exercise: Implement Sleep → Stop → Standby demos; log wake sources.
Exercise: Measure/record current for each mode
Exercise: Wake latency (µs/ms)

RTC & Tickless Timing

- LSE vs LSI trade-offs; drift & startup times; calibration basics.
- RTC calendar vs counter; wakeup timer, alarm, timestamp features.
- Using LPTIM for tickless periodic wake while main clocks are off.
- Backup registers and VBAT domain; retaining small settings across resets.
- Choosing the right kernel clock and prescalers for accuracy.

Exercise: Use LPTIM (or RTC wakeup) to wake from Stop

(Optional) UCPD — USB-C / Power Delivery

- CC pins and role detection (Sink/Source/DRP); dead-battery behavior.
- PDO negotiation basics; start with a safe 5 V sink.
- Kernel clock selection for UCPD; integration notes with external protection (TCPPx).
- Cable attach/detach events; debouncing and user feedback.
- Logging PDOs and contract changes for diagnostics.

(Optional) USB 2.0 FS Device (G0B1/G0C1)

- HSI48 + CRS crystal-less USB concept (if available); VBUS sensing.
- Endpoint/FIFO sizing; CDC vs DFU basics; LPM/remote-wakeup (device-dep.).
- Clock constraints for USB and impact on the rest of the system.
- Low-power integration while attached to USB (suspend/resume).

- Firmware update path: DFU via standard tools.

Flash, Option Bytes & Production

- Flash program/erase sequences
- Simple EEPROM emulation strategy with wear-leveling + CRC.
- Option Bytes workflow: BOR levels, boot pins, RDP levels, PCROP regions.
- Safe OB changes
- Bootloader choices (ROM vs MCUboot/SBSFU concepts)

Production Checklist

- Clocking checklist
- I/O safety at boot and sleep
- Watchdog policy + reset-cause logging; start-up self-test list
- UID/serial scheme; manufacturing data layout; versioning & image CRC
- Minimal field diagnostics: UART shell/CLI and error counters