Stand 04.04.2026

Cortex®-M7, M4, M3, M0+, M0: Arm® Cortex-M Architecture - Live Online Training

**Ziele -
Ihr Nutzen**

You know the Cortex-M7, M4, M3, M1, M0 architecture and can write software in C and Assembler. You can place the programs in memory and test them. You get the perfect introduction in developing Cortex-M based systems.

Teilnehmer

Hardware and software developers

Voraussetzungen

A basic understanding of ANSI-C and microcontrollers.

Live Online Training

20.07. – 23.07.2026	2.800,00 € *	4 Tage

* Preis je Teilnehmer, in Euro zzgl. USt.

Anmeldecode: LE-CORMX

Präsenz-Training - Englisch

Termin	Dauer
17.11. – 20.11.2026	4 Tage

Live-Online - Deutsch

Termin	Dauer
20.07. – 23.07.2026	4 Tage

Präsenz-Training - Deutsch

Termin	Dauer
17.11. – 20.11.2026	4 Tage

Cortex®-M7, M4, M3, M0+, M0: Arm® Cortex-M Architecture - Live Online Training

Inhalt

Cortex®-M (Armv7-M, Armv6-M) Processor Architecture

- Register organization, special purpose register
- Operation modes (handler/thread, privileged/unprivileged)
- Main stack, process stack
- Cortex®-M pipeline concept
- Cortex®-M memory map, system control block, bit banding

Arm Processor Cores - Overview

- Cortex®-M, Cortex®-R, Cortex®-A
- Arm7/9/10/11

Cortex®-M7, M4, M3, M0+, M0 Instruction Set

- Thumb-2 instruction set
- Data processing instructions
- Branch and control flow instructions, subroutines
- Branch table, if ... then conditional blocks
- Data access instructions
- Memory barriers and synchronization
- Exclusive access primitives
- Assembler directives
- Hands-on exercises: Generating small Assembler routines, debugging

Exception and Interrupt Handling

- Exception model
- Reset, NMI, faults, SysTick, debug, supervisor calls, external interrupts
- Tail chaining, late arriving
- Nested vector interrupt controller (NVIC)
- Interrupt configuration and status
- Interrupt prioritization, priority grouping
- Fault handler
- Hands-on exercises: SystemTick, supervisor call and PendSV in the context of RTOS applications
- Hands-on exercises: Fault handlers, output of status information

Reset Modes, Clock Generation, Power Management

- Clock generation
- Resets and Cortex®-M reset modes
- Sleep modes and power management
- System timer

Memory Protection Unit MPU for Embedded Systems

- Armv6-M and Armv7-M MPU
- Static configuration of the MPU
- Dynamic reprogramming of the MPU in an RTOS context
- Hands-on exercises: Using the MPU

Cache, Tightly Coupled Memory (TCM)

- Cache basics
- Caches and TCM of Cortex®-M7
- Cache configuration via the MPU

Embedded Core Debugging

- Core and system debugging
- JTAG debug port
- 2-pin single wire debug port
- Trace port interface unit
- Embedded trace macrocell
- Hands-on exercises: Debugging of C code using the µVision debugger and print output to the debug console

Embedded Software Development

- Adjustment of library routines to hardware (retargeting)
- Placing code and data in memory (scatter loading)
- Linker description files
- Processor start-up, start-up file

Efficient C-Programming for Cortex Architectures

- Compiler optimization, compiler options
- Interface C - Assembler
- Programming guidelines for Cortex compilers
- Optimized utilization of local and global data

Hardware-near C-Programming According to CMSIS

- Cortex Microcontroller Software Interface Standard (CMSIS)
- Software architecture for embedded systems
- Structured description of peripherals
- Access to peripherals in C
- C statements and their execution in Assembler

- Hands-on exercise: Using CMSIS functions, .e.g. for programming the NVIC interrupt controller

Floating Point Unit, Digital Signal Processing

- Architecture overview for FPU
- Exception handling using the FPU
- Single-instruction multiple data (SIMD) and saturation instructions

Overview: Cortex®-M (Armv8-M and Armv8.1-M) Processor Architecture

- Introduction: Armv8-M processor architecture
- Extensions of the Armv8.1-M processor architecture (HELIUM)
- Difference to the Armv6-M and Armv7-M processor architecture
- Differences of the new Armv8-M MPU
- Overview: Cortex®-M23, M33 and Arm TrustZone

Exercises with Keil µVision in Assembler and C

- Hands-on exercises on Armv7-M Cortex-M7 are developed and tested on STM32F746 Nucleo board.
- The exercises are done using Keil Studio (Visual Studio Code). Keil uVision is sometimes used as a debugger.

MicroConsult PLUS:

- We will provide you with a download link for your exercise directory and sample solutions for all exercises.
- In addition, you get installation instructions with download links for the tool environment for you to reproduce the exercises after the training.