# Arm-Cortex-R-Software Development

## Summary

Arm Cortex-R software training courses are designed to help engineers working on new or existing Cortex-R system designs. Whether you're working on design, verification or validation, for a Cortex-R system, the course can be **configured according to your team's needs**.

Courses include fundamental topics to enable a solid platform of understanding. The rest of the course then builds on from this with optional topics and can be tailored appropriately. Some key topics are delivered via **pre course on-demand video**.

**Learning activities** such as interactive workbooks, walkthrough examples and quizzes are incorporated into the training to help bring the learning to life.

A **pre course call** with the engineer delivering the training will help you discuss your team's individual training requirements.

At the end of the course delegates will be able to:

- Describe different Cortex-R processors features and their use.
- Understand the programmer's model of a Cortex-R processor.
- Identify and solve key Cortex-R system design issues.
- Program simple bare-metal code in both C and Arm assembly language.
- Debug issues on Cortex-R processors.

| Course Length | Delivery Method | Location |
|---|---|---|
| 3 – 4 days | Classroom | Virtual or Onsite |

## Audience

- System architects
- Real-time operating system developers
- Device driver developers
- Low level software developers
- Engineers writing low level test code

## Prerequisites

- A basic understanding of microprocessor systems
- Familiarity with assembler or C programming
- Experience of embedded system development is helpful but not essential
- A basic awareness of Arm is an advantage but not required

## Related Products

Armv7-R, Armv8-R, Cortex-R4, Cortex-R5, Cortex-R7, Cortex-R8, Cortex-R52, Cortex-R52+, Cortex-R82

## Topics

Agendas will be created from the following list of fundamental and optional topics

### Fundamental Topics

- **Introduction to the Arm Architecture** and feature set of your chosen Cortex-R processor. A discussion of the programmer's model, register layout and architectural features. ♥
- **Assembly programming**. Introducing the instruction set and assembly directive available and how to use them.
- A discussion of the **interrupt** and **exception** architecture, including how to handle IRQs and internal faults and how to program the interrupt controller.
- **Cortex-R memory model,** covering Arm memory types, interactions with caches and Tightly Coupled Memory (TCM) and how to program the Memory Protection Unit (MPU.)
- An advanced discussion of Arm memory accesses, including **memory barrier instructions** and **Load/Store Exclusive** instructions for inter-process **synchronisation**.
- A discussion of the effective use of compilation tools with a Cortex-R system. Covering writing effective and efficient **C code** and basic **linker layout**. Including the bare metal software **boot flow from reset to C main().**
- **Embedded virtualization**. Interrupt virtualization, instruction trap-and-emulate and the two stage MPU. (Cortex-R52 only.)

### Optional Topics

- **NEON overview** covering the Single Instruction Multiple Data (SIMD) instructions available for Cortex-R processors. (Cortex-R52 only.)
- **Debug**. A discussion of the Cortex-R debug architecture focussing on the low level feature that enable a debugger to connect to and debug your CPU.

♥ = Online and on-demand.