Arm-Cortex-R-Hardware Design

Summary
Arm Cortex-R Hardware training courses are designed to help engineers working on new or existing Cortex-R system designs. Whether you’re working on design integration or verification, for a Cortex-R system, the course can be configured according to your team’s experience and relevant Arm IP.

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.

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.
• Identify and solve key Cortex-R system design issues.
• Make appropriate system design choices.
• Decide on the best configuration options for their system.
• Debug simulation issues on Cortex-R processors.

<table>
<thead>
<tr>
<th>Course Length</th>
<th>Delivery Method</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 – 4 days</td>
<td>Classroom</td>
<td>Virtual or Onsite</td>
</tr>
</tbody>
</table>

Audience
• System designers
• RTL integration team
• RTL verification team
• Physical implementation team*

*Topics applicable to physical implementation team are limited, please discuss this with Arm if you have physical implementation training requirements

Prerequisites
• Basic understanding of Verilog
• Basic programming experience (but it does not have to be ARM programming)

Related Products
Armv7-R, Armv8-R, Cortex-R4, Cortex-R5, Cortex-R7, Cortex-R8, Cortex-R52
Topics
Agendas will be created from the following list of fundamental and processor specific topics

**Fundamental Topics**

- **Introduction to the basics of the Arm Architecture and Assembly programming.** A discussion of the programmer’s model, register layout and architectural features and the basics of the instruction set.
- **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.)
- How the Arm architecture requirements translate into the design of a Cortex-R processor and therefore the typical Cortex-R processor behaviours will cause interaction with a system.
- The implication of shared memory and devices in the system and its requirements for memory access **Coordination** and **cache coherency** for the Cortex-R processor.
- Basic requirements of the **AXI** protocol used for the main Cortex-R processor interfaces.

**Processor Specific Topics**

- The basic **instruction execution** behaviour of the processor.
- The **memory access** behaviour considering both the innate behaviour and the hardware control and configuration available.
- The **fault protection** capability built into the processor and how it is intended to be used in a system.
- The interaction between the processor and the system, and the expected connectivity and controls for the **system interaction**.
- The behaviour and connectivity of the **interrupt controller** for the processor.
- The **debug capability** behaviour for the processor and how it interacts with the system **debug infrastructure**.
- The design and system control of the processor **clock, reset and power management**.
- **Configuration and implementation flow** that is expected to be used to take the RTL netlist through to a physical implementation highlighting any key requirements for configuration and implementation decisions.
- Example Assembly programming required for a **basic processor boot** suitable for verification simulations.

**Related face-to-face and on-demand courses**

- CoreSight Training