RR3 - ARM Cortex-R52/R52+ Implementation and software design

Objectives

- Understand ARM v8-R architecture and Cortex-R52/R52+ features.
- Learn pipeline behavior and instruction execution.
- Master exception handling and memory systems.
- Explore virtualization and safety features.
- Implement synchronization and debug techniques.

Prerequisites

- Basic knowledge of ARM architecture
- Familiarity with embedded systems
- Experience in assembler programming (optional)

Target audience

- Software developers working with ARM architecture
- System architects
- Embedded systems engineers

Course Environment

- Theoretical course
  - PDF course material (in English) supplemented by a printed version.
  - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Course Outline

First Day

ARM v8-R Architecture overview

- Core registers
- Exception model
- Instruction sets
- Memory model
- Virtualization

Introduction to Cortex-R52/R52+

- Overview
- Memory System
  - TCM Memory
  - Level-1 caches
  - Direct access to internal memory
  - AXIM interface
  - Low-latency peripheral port

    - Flash interface
    - AXIS interface
    - Error detection and handling
- Safety and Configurable options
    - DCLS
    - Spin-lock

Pipeline and ISA

- Pipeline
- Instruction execution
- Conditional instructions
- Flag-setting instructions
- Timings
    - Instructions cycle timings
    - Base instructions cycle timings
    - Pipeline Behavior
    - Skewing
    - Dual-issuing
    - Load / Store
    - Division and square root
    - Floating-point and Advanced SIMD Multiply accumulate instructions
    - Instructions with exceptional behavior
- A32 and T32 instructions

Second Day

Exceptions Model

- Exception state
- Exception levels
- Reset state in ARMv8-R
- Interrupt
    - Controller
    - Handling
- Virtualization

Level 1 memory system (Cache and TCM)

- Cache
    - Cache basics: organization, replacement algorithm, write policies
    - Cache organization
    - Write with allocate policy
    - Understanding transient cache line load / store: linefill buffers, eviction buffer
    - Cache maintenance operations
- TCM
    - Tightly Coupled Memories, address decoding
    - ITCM and DTCM configuration
    - Accessing the TCMs from the AXI slave interface
    - ECC protection, TCM internal error detection and correction
    - Preloading TCMs with ECC
    - Using TCMs from reset
- Memory ordering
- Memory Barriers
- Shared Resource management

Memory Protection Unit (MPU)

- Memory protection overview
- MPU regions
- Virtualization support

Third Day

GICv3

- Introduction
- Fundamentals
- Configuring the GIC (SPI, PPI, SGI …)
- Handling Interrupts
- Configuring LPIs
- Virtualization

ARMv8-R Virtualization with safety considerations

- Introduction to virtualization
  - Basic concepts
  - Virtual interrupts
  - Trapping exceptions
  - Trapping instructions and register access
  - PMSA (MPU)
  - Virtual timers
  - Virtual machine IDs
  - Backup registers
- Examples of virtualization
  - Guest OS switcher
  - OS monitor
  - Virtual interrupts

Synchronization overview

- Inter-Processor Interrupts
- Cluster ID
- Exclusive access monitor, implementing Boolean semaphores
- Global monitor
- Spin-lock implementation
- Using events

Performance Monitoring Unit (PMU)

- Event interface
- Counters
- Authentication signals and PMU behavior

Debug

- Debug overview (External debug, self-hosted debug …)
- Cross trigger
- Embedded Trace Macrocell (ETM)