AAM - ARM Cortex-M Architecture (v7/v8)

This course explains the ARM Cortex-M global architecture.

Objectives

- Describing the ARM Cortex-M processors architecture (ARMv6, ARMv7 and ARMv8)
- Presenting the Hardware and Software implementation possibilities to learn how to create Cortex-M based applications
- Review the differences between the different Cortex-M cores
- Get an overview of the new features included in ARMv8 architecture (TrustZone, MPU, New Memory Types, ...)
- Learn how to:
  - Develop and debug an ARM Cortex-M application
  - Configure and Manage Exceptions/Interrupts
  - How to configure a privileged and unprivileged access with the MPU
- This course provides all the prerequisites for the courses describing in details the various Cortex-M cores and CPUs.

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

Prerequisites

- Familiarity with embedded C concepts and programming
- Basic knowledge of embedded processors

Target Audience

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

Evaluation modalities

- The prerequisites indicated above are assessed before the training by the technical supervision of the traineein his company, or by the trainee himself in the exceptional case of an individual trainee.
- Trainee progress is assessed in two different ways, depending on the course:
  - For courses lending themselves to practical exercises, the results of the exercises are checked by the trainer while, if necessary, helping trainees to carry them out by providing additional details.
  - Quizzes are offered at the end of sections that do not include practical exercises to verifythat the trainees have assimilated the points presented
- At the end of the training, each trainee receives a certificate attesting that they have successfully completed the course.
  - In the event of a problem, discovered during the course, due to a lack of prerequisites by the trainee a different or additional training is offered to them, generally to reinforce their prerequisites,in agreement with their company manager if applicable.

Plan

First Day

ARM v6M/v7M/v8M Architecture Overview

- Introduction to the ARM Architecture
- Cortex-M Processors
- Programmers' Model
  - Core Registers
  - Privileges, Modes and Stacks
  - Instruction Set
  - Datapath and pipeline, speculative branch target prefetch
- Exception Model
- Memory Model
  - Address Map
  - Memory Types
  - Instruction and data alignment
  - System Control Space
- Power Management

Exercise: *Core register review and changing Mode and Stack*
Exercise: *Using low power mode*

Cortex-M Implementation Diversity

- ARM Cortex-M0 processor
- ARM Cortex-M0+ processor
- ARM Cortex-M3 processor
- ARM Cortex-M4 processor
- ARM Cortex-M7 processor
- ARM Cortex-M33 processor
- ARM Cortex-M23 processor
- ARM Cortex-M processor family comparison

Second Day

Cortex-M Software Development

- Tools
  - Keil IDE and Ulink2 probe presentation
  - System Workbench for MCU
- Standards

    - AAPCS
    - UAL
    - CMSIS
- Code Generation
    - Variable types supported
    - Register Usage, Parameter passing
    - Aligned and Unaligned accesses
    - Endianness
- Image Generation
    - Dealing with branches
- Fault Tolerance
    - Stack issues
- Determinism
- RTOS Support
    - MPU Overview
    - Systick Timer Overview

Exercise: *AAPCS review, CMSIS utilization and Assembly language inlining*

Cortex-M Optimization

- Compiler optimizations
    - Optimization levels
    - Tail-call
    - Inlining of function
    - Loop transformation
    - Mutlifile compilation
    - Floating point
- Bit Banding
- Memory copy optimizations
- Base pointer optimization

Exercise: *Bit banding implementation*

Cortex-M Debug

- ARMv6-M and ARMv7-M Debug Overview
    - Coresight presentation
- Invasive Debug
    - Beakpoints and Watchpoints
    - Vector Catch
    - Semi-hosting
- Non-invasive Debug
    - Data Watchpoint and trace unit
    - Instrumentation Trace Macrocell (ITM)
    - Embedded Trace Macrocell (ETM)
    - Micro Trace Buffer (MTB)

Exercise: *Debug features review through the Keil IDE*

Cortex-M Startup and Linker

- Reset Behavior
    - Vector Table
- CMSIS-CORE Startup and System Initialization
    - Startup File
    - Exception Handlers
    - Stack and heap setup
- Post Startup Initialization
- Working with the linker
    - Creating code and data sections
    - Placing code and data in memory

Exercise: *Startup sequence to the main() review*
Exercise: *Executing the code from a SRAM*

Third Day

Cortex-M Exception Model

- Exception Handling
- Exception entry and exit
- Exception stacking
- Nesting
- Tail-chaining
- Late-arriving

Exercise: *Exception entry review*

- Prioritization and Control
- NVIC registers
- Priority boosting
- Priority grouping
- Masking exceptions
- Writing Interrupt Handlers
- Interrupt Sensitivity
- Internal Exceptions and Faults
- Fault escalation

Exercise: *Managing interrupts and priorities*

Memory Protection Unit (MPU) ARMv7 and ARMv8

- MPU regions
- Privileged vs Unprivileged
- Memory Types
- Access permissions
- Region overlapping
- Access Fault

Exercise: *MPU Utilization*

Fourth Day

Cortex-M TrustZone

- Security States
- Register Banking
- Secure State Address Protection
- Secure and Non-Secure Interactions

Cortex-M Advanced Features

- SysTick Configuration and Calibration

Exercise: *Working with the Systick Timer*

- Synchronization
  - Critical section, atomicity
  - LDREX/STREX instructions
  - Lock and unlock examples
- Memory Barriers
  - Data memory, Data Synchronization and Instruction Synchronization Barriers
  - Utilization examples
- Further Instruction Set Information
  - Return on the Instruction Set
  - If-Then Block
  - DSP extension Overview

Exercise: *Using DSP instructions*

Renseignements pratiques

Inquiry : 4 days