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

Ce cours explique l'architecture globale ARM Cortex-M

Objectifs

- Description de l'architecture des processeurs ARM Cortex-M (ARMv6, ARMv7 et ARMv8).
- Passer en revue les différences entre les différents noyaux Cortex-M.
- Obtenez un aperçu des nouvelles fonctionnalités incluses dans l'architecture ARMv8 (TrustZone, MPU, nouveaux types de mémoire, ...).
- Apprendre à :
  - Développer et déboguer une application ARM Cortex-M
  - Configurer et gérer les exceptions/interruptions
  - Comment configurer un accès privilégié et non privilégié avec le MPU.
- Ce cours fournit tous les prérequis pour les cours décrivant en détail les différents cŒurs et CPU Cortex-M.

Environnement du cours

- Cours théorique
  - Support de cours au format PDF (en anglais) et une version imprimée lors des sessions en présentiel
  - Cours dispensé via le système de visioconférence Teams (si à distance)
  - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique
- Activités pratiques
  - Les activités pratiques représentent de 40% à 50% de la durée du cours
  - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
  - Exemples de code, exercices et solutions
  - Pour les formations à distance:
    - Un PC Linux en ligne par stagiaire pour les activités pratiques, avec tous les logiciels nécessaires préinstallés.
    - Le formateur a accès aux PC en ligne des stagiaires pour l'assistance technique et pédagogique
    - Certains travaux pratiques peuvent être réalisés entre les sessions et sont vérifiés par le formateur lors de la session suivante.
  - Pour les formations en présentiel::
    - Un PC (Linux ou Windows) pour les activités pratiques avec, si approprié, une carte cible embarquée.
    - Un PC par binôme de stagiaires s'il y a plus de 6 stagiaires.
  - Pour les formations sur site:
    - Un manuel d'installation est fourni pour permettre de préinstaller les logiciels nécessaires.
    - Le formateur vient avec les cartes cible nécessaires (et les remporte à la fin de la formation).
- Une machine virtuelle préconfigurée téléchargeable pour refaire les activités pratiques après le cours
- Au début de chaque session (demi-journée en présentiel) une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

Pré-requis

- Familiarité avec les concepts et la programmation du C embarqué
- Connaissance de base des processeurs embarqués

Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

Modalités d'évaluation

- Les prérequis indiqués ci-dessus sont évalués avant la formation par l'encadrement technique du stagiaire dans son entreprise, ou par le stagiaire lui-même dans le cas exceptionnel d'un stagiaire individuel.
- Les progrès des stagiaires sont évalués de deux façons différentes, suivant le cours:
  - Pour les cours se prêtant à des exercices pratiques, les résultats des exercices sont vérifiés par le formateur, qui aide si nécessaire les stagiaires à les réaliser en apportant des précisions supplémentaires.
  - Des quizz sont proposés en fin des sections ne comportant pas d'exercices pratiques pour vérifier que les stagiaires ont assimilé les points présentés
- En fin de formation, chaque stagiaire reçoit une attestation et un certificat attestant qu'il a suivi le cours avec succès.
  - En cas de problème dû à un manque de prérequis de la part du stagiaire, constaté lors de la formation, une formation différente ou complémentaire lui est proposée, en général pour conforter ses prérequis, en accord avec son responsable en entreprise le cas échéant.

Plan

Premier jour

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

Deuxième jour

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*

Troisième jour

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*

Quatrième Jour

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

Renseignements : 4 jours