RM5 - Cortex-M33 Implementation

This course covers the Cortex-M33 ARMv8 core

Course objectives

- This course is split into 3 parts:
  - Cortex-M33 architecture
  - Cortex-M33 software implementation and debug
  - Cortex-M33 DSP programming.
- The Cortex-M33, although a 32-bit core, is one of the first ARMv8-M cores proposed by ARM and includes several advanced features.
- The Cortex-M33 low level programming is explained, particularly the ARM linker parameterizing and some tricky assembly instructions.
- The course also indicates how to the DSP and FPU instructions to boost DSP algorithm implementation.
- The various Coresight debug elements implemented in the Cortex-M33 are also presented

A more detailed course description is available on request at training@ac6-training.com

Prerequisites

- Basic understanding of microprocessors and microcontrollers

Course Environment

- Labs will be executed on an ARMv8 simulator.
- Printed training material is given to attendees during training.
- Precise and easy to use, it can be used as a reference afterwards.

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

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

First Day

Introduction to ARMv8-M Architecture

- ARM Cortex-M33 processor macrocell
- ARMv8-M Programmer’s model
- Instruction pipeline
- Fixed memory map
- Privilege, modes and stacks
- Memory Protection Unit
- Security extensions
- Interrupt handling
- Nested Vectored Interrupt Controller [NVIC]
- Power management
- Debug

ARM Cortex-M33 core

- Special purpose registers
- Datapath and pipeline
- Write buffer
- Bit-banding
- System timer
- State, privilege and stacks
- System control block

Architecture of a SoC based on Cortex-M33

- Internal bus matrix
- External bus matrix to support DMA masters
- Connecting peripherals
- Sharing resources between Cortex-M4 and other CPUs

- Connection to Power Manager Controller

Second day

Embedded Software Development with Cortex-M33

- Application startup
- Placing code, data, stack and heap in the memory map, scatterloading
- Reset and initialisation
- Placing a minimal vector table
- Further memory map considerations, 8-byte stack alignment in handlers

Exercise : *Create a standalone C application displaying data on a serial line*

The T32 Instruction Set variant supported on ARMv8M

- General points on syntax
- Data processing instructions
- Branch and control flow instructions
- Memory access instructions
- Exception generating instructions
- If…then conditional blocks
- Stack in operation
- Stack limit registers
- Exclusive load and store instructions, implementing atomic sequences
- Memory barriers and synchronization

Exercise : *Create assembly-level functions to implement simple algorithms*

Synchronization and Semaphores

- Exclusive access instructions
- The Local, Global and External monitors
- Interaction with exclusive access instructions
- Load Exclusive and Store Exclusive usage and constraints

Exercise : *Implement atomic variable manipulation using exclusive access instructions*
Exercise : *Implement spinlocks*

Cortex-M DSP Instruction Set

- Multiply instructions
- Packing / unpacking instructions
- V6 ARM SIMD packed add / sub instructions
- SIMD combined add/sub instructions, implementing canonical complex operations
- Multiply and multiply accumulate instructions
- SIMD sum absolute difference instructions
- SIMD select instruction
- Saturation instructions

Exercise : *Code assembly-language optimized data-processing algorithms*

Third day

CMSIS DSP support

- The CMSIS library framework
- The CMSIS DSP intrinsic functions
- The optimized CMSIS data-processing functions

Exercise : *Recode data-processing functions in C using intrinsics*
Exercise : *Recode the same using CMSIS high-level data processing functions*
Exercise : *Compare performance of the various implementations*

Floating point Unit

- Introduction to IEEE754
- Floating point arithmetic
- Cortex-M4F single precision FPU
- Register bank
- Enabling the FPU
- FPU performance, fused MAC
- Improving the performance by selection flush-to-zero mode and default NaN mode
- Extension of AAPCS to include FP registers

Exercise : *Enable the FPU and use it for simple floating point algorithms*

C/C++ Compiler hints and Tips for Cortex-M33

- Mixing C/C++ and assembly
- Coding with GCC compiler
- Measuring stack usage
- Unaligned accesses
- Local and global data issues, alignment of structures
- Further optimisations, linker feedback

Exercise : *Measure stack usage of a program*
Exercise : *Place a user-defined data structure at a fixed address*

Exceptions

- Exception behavior, exception return
- Non-maskable exceptions
- Privilege, modes and stacks
- Fault escalation
- Priority boosting
- Vector table

Exercise : *Manage synchronous exceptions to simplify FPU usage*
Exercise : *Manage the SVC exception to switch between user and privileged modes*

Fourth day

Interrupts

- Basic interrupt operation, micro-coded interrupt mechanism
  - Interrupt entry / exit, timing diagrams
  - Interrupt stack
  - Tail chaining
- Interrupt response, pre-emption
- Interrupt prioritization
- Interrupt handlers
- The Nested Vectored Interrupt Controller (NVIC)

Exercise : *Handle a timer interrupt in C or assembly language*
Exercise : *Manage interrupt masking and nesting between two interrupts*

The Security Extension

- Security states
- Register banking between security states
- Stacks and security states
- Security Extension and exceptions
- Secure state address protection
- Secure and Non-Secure states interactions
  - Secure sate transitions

  - Function calls from Non-Secure to Secure state
  - Returning from Secure state
- Exceptions and the Security Extension
  - Handling Secure Exceptions
  - Handling Non-Secure Exceptions while in the Secure state
  - Returning from a Non-Secure exception to the Secure state
- The Security Attribution Unit
- The Implementation Defined Attribution Unit

Exercise : *Implement a minimal secure monitor*

Memory Protection Unit

- Memory types
- Access order
- Memory barriers, self-modifying code
- Memory protection overview, ARM v8 PMSA
- Cortex-M33 MPU and bus faults
- Fault status and address registers
- Region overview, memory type and access control
- Setting up the MPU

Exercise : *Use the MPU to protect an area of memory against unintended access*

Debugging features

- Invasive Debug
  - Coresight debug infrastructure
  - Halt mode
  - Vector catching
  - Debug event sources
  - Flash patch and breakpoint features
  - Data watchpoint and trace
  - ARM debug interface specification
  - Coresight components
  - AHB-Access Port
  - Possible DP implementations: Serial Wire JTAG Debug Port [SWJ-DP] or SW-DP
- Non-Invasive debug
  - Basic ETM operation
  - Instruction trace principles
  - Instrumentation trace macrocell
  - ITM stimulus port registers
  - DWT trace packets
  - Hardware event types
  - Instruction tracing
  - Synchronization packets
  - Interface between on-chip trace data from ETM and Instrumentation Trace Macrocell [ITM]
  - TPIU components
  - Serial Wire connection

Renseignements pratiques

Durée : 4 jours
Prix : 2860 € HT