Formation Architecture ARM Cortex-A et R: Ce cours explique l'architecture globale ARM Cortex-A et R - Processors: ARM Cores

AAA - Architecture ARM Cortex-A et R

Ce cours explique l'architecture globale ARM Cortex-A et R

Objectifs

- Description des profils d'architecture ARM v7 et v8 A et R
- Description des différentes architectures de processeurs ARM Cortex-A et R
- Présentation des possibilités de mise en œuvre matérielle et logicielle pour apprendre à créer des applications basées sur le Cortex-A
- Détailler les fonctionnalités de sécurité (TrustZone) et de virtualisation (Hypervisor) de l'architecture ARMv8
- Ce cours fournit tous les prérequis pour les cours décrivant en détail les différents cœurs et processeurs Cortex-A et Cortex-R

Environnement du cours

- Support de cours pratique avec espace pour la prise de notes
- Démonstrations ciblant un SoC basé sur Cortex-A9

Pré-requis

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

Plan

Premier jour

Architecture

- Introduction to ARM and the Architecture
- The AArch32 Programmer’s Model
- The AArch64 Programmer's Model
- Exceptions
- Memory Architecture
- Caches

Implementations

- Versions and Implementations
- ARMv4T
- ARMv5TE

- ARMv6
- ARMv7
- ARMv8
- SecurCore
- Architecture Extensions
- Pipelines
- Cycle Counting

Deuxième jour

System features

- Multi-processing
- Cache maintenance
- Cache coherency hardware
- Interrupt distribution
- Power saving modes
- Memory system hierarchy
- Software storage and upload

AArch64 Exception Model

- Four exception levels
- Exception Link Registers
- Register banking by exception level
- Nesting on the same exception level
- Exception type and exception origin
- Syndrome registers used to provide status information to the exception handler
- Exception return instruction
- AArch64 Exception vector tables

Generic Interrupt Controller

- Generic Interrupt Controller CPU Interface Registers
- Interrupt Virtualization
- Interrupt Handling to support Nesting

Troisième jour

Multicore operation

- Single Processor / Multi-Task RTOS
- Multi-CPU Exclusive Resource Management
- Wait for Event / send Event
- Wait for Interrupt
- Multi-Processor / Multi-Task RTOS

Software Development

- Embedded Software Development
- Libraries and Linkage
- Target platforms
- Memory ordering models
- Barriers and synchronization

- Cache policies
- Operating system support
- Booting

Software Optimization

- Introduction
- Coding techniques
- Profiling

Software debug

- Debug basics
- Debug hardware
- Invasive Debug
- Non-invasive Debug
- Standard Debug Techniques
- Timing
- Resources

CoreSight Debug Components

- Self-Hosted Debug
- Debug State Instructions
- Linked comparisons for Breakpoint/Watchpoint exception generation
- Software Step exceptions
- Routing debug exceptions
- External debug, cross-triggering
- Embedded Trace Macrocell architecture

Quatrième jour

ARMv8 Memory Management Unit

- ARMv7 MMU and LPAE compatibility
- LPAE enhancements to adapt to AArch64
- Supporting up to 48 bits of VA per TTBR
- Access permission checking
- Supporting up to 48 bits of IPA and PA spaces
- VMSAv8-64 address translation system
- Memory translation granule size
- Descriptor page table organization, descriptor format
- Hierarchical control of Secure or Non-secure memory accesses

The ARMv8 Security Model

- Compatibility with ARMv7
- Security model when EL3 is using AArch64
- Trapping to EL3 using AArch64
- Re-entrant mode
- Secure exception management, trapping
- Asynchronous exception routing and control

Virtualization

- New hypervisor privilege level on non-secure side

- Re-entrant mode
- Virtualization Extension Effect on MMU
  - Second stage MMU
  - I/O MMU
  - Managing external masters programmed by the guest OS without an I/O MMU
- Emulation support
- Hypervisor exception management, trapping
- Asynchronous exception routing and control
- Resource management
- Virtualization modes
  - Para virtualization versus full virtualization
  - Separation kernels
  - Partitioning kernels
  - Operating-system virtualization (containers)
  - Existing hypervisors (Xen, KVM…)

Renseignements pratiques

Durée : 4 jours
Prix : 2200 € HT