Cortex-A9 MPCore Software Development

Course Description

Cortex-A9 MPCore software development is a 4 days ARM official course. The course goes into great depth and provides all necessary know-how to develop software for systems based on Cortex-A9 processor.

The course covers the Cortex-A9 MPCore architecture, instruction set, exception handling, memory management unit, cache and branch prediction, processes synchronization, C programming, cache coherency, boot process, barriers, NEON coprocessor, power management, debug and security.

At the end of the course the participant will receive a certificate from ARM.

Course Duration

4 days (5 days with labs)

Goals

1. Become familiar with ARMv7-A architecture
2. Become familiar with Cortex-A9 MPCore architecture
3. Become familiar with ARMv7-A instruction set
4. Understand the exception handling mechanism
5. Be familiar with Cortex-A9 caches and maintenance operations
6. Be able to configure and use the MMU
7. Write an efficient C code for Cortex-A processor
8. Be able to boot Cortex-A9 MPCore system
9. Implement synchronization processes using mutex/semaphore
10. Be able to add barriers instructions to control program flow
11. Be able to program the GIC
12. Become familiar with NEON coprocessor SIMD capabilities
13. Manage Cortex-A9 MPCore power modes
14. Be able to debug with invasive and non-invasive techniques
15. Become familiar with TrustZone infrastructure to build secured systems
16. Embed AMP and SMP operating systems

Target Audience

Software engineers that would like developing software and BSP for platforms based on Cortex-A9 MPCore processor.

Prerequisites

- Computer architecture background
- C and Assembler
- Experience in developing embedded systems

Course Material

- ARM official course book
- Labs handbook
- DS5 SDK

Agenda

Main Topics:

- ARM Processor Cores
- ARM System Design
- Introduction to the ARM Architecture
- ISA Assembly
- Exception Handling
- Cortex-A9 MPCore Overview
- Caches and Branch Prediction
- Using the MMU
- Writing C for ARM
- Booting a Cortex-A9 MPCore
- Barriers
- OS support
- Synchronization
- Cache coherency
- Programming the GIC
- Cortex-A9 power management
- Debug and trace
- Neon Overview
- Introduction to TrustZone

Day #1

❖ Introduction to the ARM architecture

- Architecture Versions
  - Introduction to the ARM architecture
  - Development of the ARM architecture
  - ARM Cortex processors (A/R/M)

- Registers & Instruction Sets
  - Data sizes and instruction sets
  - The ARM register set
  - Program status register
  - ARM, Thumb. Thumb2, ThumbEE, Jazelle
  - Endianness
  - Assembler syntax examples
  - Floating point and NEON
  - AAPCS

- Exception Model
  - Processor modes
  - Banking of registers
  - Taking an exception
  - Vector table

- Memory Model
  - Memory model overview
  - Memory types (Normal/Device/Strongly Ordered)
  - Memory hierarchy example
  - Data alignment

- Coprocessors
  - Coprocessors overview
  - CP15 example
  - PMU

- Architecture Extensions
  - TrustZone
  - Virtualization
  - Jazelle

❖ **ARMv7-A ISA Overview**

- ARM Assembler File Syntax
- Load/Store Instructions
  - Single/Double register data transfer
  - Addressing memory
  - Pre and post-indexed addressing
  - Multiple register data transfer

- Data Processing Instructions
  - Arithmetic, logical, move instructions
  - Shift/Rotate operations
  - The flexible second operand
  - Instructions for loading constants
  - Multiply/Divide
  - Bit manipulation instructions
  - Byte reversal

- Flow Control Instructions
  - Branch instructions
  - Interworking
  - Compare and branch if Zero
  - Condition codes and flags
  - If-Then instruction
  - Supervisor call instruction (SVC)

- Miscellaneous Instructions
  - Coprocessor instructions
  - PSR access
  - Breakpoint instruction (BKPT)
  - Wait for interrupt instruction (WFI)
  - NOP instruction
  - Wait for event & send event instructions (WFE & SEV)

- DSP Instructions
  - SIMD
  - Saturated maths and CLZ
  - Data packing/unpacking

❖ **Exception Handling in Details**

- Introduction
  - Exception handling process
  - The ARM register set and modes
  - Exception priorities
  - Vector table
  - Link register adjustments

- Returning from exceptions
- Exception state & Endianness
- Non-maskable fast interrupt
- Low latency interrupt

- Interrupts
  - Interrupt & interrupt handler example
  - Interrupt pre-emption
  - Issues with re-enabling interrupts
  - Change processor state (CPS) instruction
  - Stack issues
  - Nested interrupt example
  - FIQ vs IRQ
  - Interrupt controllers

- Abort Handlers
  - Prefetch and data aborts
  - Data abort types (internal/external, precise/imprecise)
  - Identifying the abort source
  - Example data abort handler

- SVC Handlers
  - What is SVC used for?
  - Example SVC handler
  - Example SVC usage in an OS

- Undef Handlers
  - Undefined instruction
  - Example Undef handler

- Reset Handlers

Day #2

❖ Cortex-A9 MPCore Architecture

- Software Engineer's Guide to the Cortex-A9
  - Cortex-A9 architecture overview
  - Cortex-A9 pipeline
  - Cortex-A9 MPE configuration
  - Register renaming
  - Out of order issue/completion
  - Small loop mode
  - Program flow prediction
  - Performance Monitoring Unit (PMU)
  - Level 1 memory system (caches and MMU)
  - Prefetching and preload

❖ Caches & Branch Prediction

- ARMv7-A Caches and Branch Prediction
  - Caches in Cortex-A series processors
  - Level 1 and level 2 cache interaction
  - Inner and outer cache policies
  - Speculative prefetch and preload
  - L1 memory system buffers
  - When should I enable caches?
  - Non-deterministic cache behavior
  - Cache maintenance and coherency
  - Branch prediction

❖ MMU

- Memory Management Unit
  - MMU overview
  - Short-descriptor format
    - Level one and level two tables
    - First level translation table
    - First level short descriptor format
    - Second level translation table
    - Second level short descriptor format
  - Long-descriptor format
    - Memory types and attributes
    - Memory types (normal, device, strongly ordered)
    - Memory type access order
    - Instruction accesses
    - Hierarchical attributes

- Using the MMU
  - MMU registers
  - Enabling the MMU
  - Memory faults
  - Synchronous and asynchronous aborts
  - Security extensions

❖ **C Programming & Boot**

➢ Writing C for ARM
- Parameter passing
  - Parameter passing
  - Passing more than 4 parameters
  - Parameter alignment
- Floating point linkage
  - HW and SW floating point linkage
  - Floating point linkage example
- Alignment
  - Global data layout
  - Unaligned accesses
  - Packing and alignment of structures
  - Alignment of pointers
- Coding considerations
  - Size of local variables
  - Division
  - Base pointer optimization
  - Using “volatile”

➢ Booting a Cortex-A9 MPCore
- Overview
  - Booting considerations
  - Bare metal vs OS
  - Warm vs cold reset
  - Overview of cold boot process
  - Overview of warm reset process
- Booting a single CPU
  - Reset state
  - Vector table initialization
  - Stack initialization
  - Basic memory system initialization
  - VFP/NEON initialization
  - TrustZone

- Booting a cluster

Day #3

❖ Understanding Barriers

- Overview
  - Memory model
  - Why do I care about access order?
  - Barriers (DMB, DSB, ISB)
- Data Barriers
  - DMB vs DSB
  - DMB instruction example
  - DSB instruction example
  - Mail box example
  - Speculation across barriers
- Instruction Barriers
  - ISB instruction
  - CP15 example
  - Translation table change example
  - Self-modifying code example
- Compiler Barriers

❖ OS Support

- Multi-Processing
  - What is multi-processing?
  - Symmetric Multi-Processing (SMP)
  - Asymmetric Multi-Processing (AMP)
  - Which CPU am I?
  - Cortex-A9 private memory region
  - Sharing translation tables
- Translation Tables
  - Dual translation tables
  - Unused memory
  - Translation table change example
  - Translation table memory use
  - Long-descriptor page table sizes
  - Caching translation tables

- TLB entries

- Context Switching
  - Context
  - Address Space Identifiers (ASIDs)
  - Thread switching using ASIDs
  - ASID with short-descriptor format
  - Other ASID considerations
  - Exclusive monitor
  - Migrating a thread across CPUs
  - VFP/NEON

- Timers
  - Timers in an MP system

❖ **Synchronization**

- The Need for Atomicity
- The Race for Atomicity
- Critical Sections
- Effective Atomicity
- LDREX and STREX Instructions
- Example… lock() and unlock()
- Programs Still Have to be Smart
- Multi-Thread Mutex Example
- Coherent Multi-Core
- Synchronization in a Cluster Example
- Non-Coherent Multi-Core
- Memory Attributes
- Context Switching
- Exclusive Reservation Granule

❖ **Cortex-A9 Cache Coherency**

- L1 Cache Coherency and Maintenance
  - Cache maintenance
  - Coherency operations
  - Point of Unification (PoU)
  - Point of Coherency (PoC)
  - PoU vs PoC
  - Ordering of maintenance operations
  - Ordering between I/D/Table walk
  - Maintenance operations list

- MPCore Coherency
  - Coherency management
  - Maintenance operations broadcast

- Accelerator Coherency Port (ACP)
- Coherency example
- Coherent transactions using MESI

❖ **Generic Interrupt Controller Programming**

- GIC Overview
  - GIC architecture
  - Sources of interrupt (SGI, PPI, SPI)

- Distributor and CPU Interfaces
  - Register interfaces
  - Distributor interface
  - CPU interface
  - Programming guidelines

- How to Enable and Configure Interrupts
  - Enabling the IC
  - Interrupt configuration

- How to Handle Interrupts
  - Interrupt states
  - Taking an interrupt
  - Which CPU services an SPI?
  - Priority mask register
  - Interrupt priority registers
  - Pre-emption
  - Nesting interrupts

- How to Send Software Interrupts
  - SGI capability
  - Sending a SGI
  - Receiving a SGI

- Security Extensions
  - Group 0 and Group 1
  - Acknowledging interrupts
  - Priority and banking

- Interrupts IDs on Cortex-A9

Day #4

❖ Power Management for Cortex-A Processors

- Power Overview
  - Power consumption
  - Example power contributions
  - Power reduction techniques
  - Example power domains
  - Additional power modes/interconnect
  - Energy cost

- Processor Power Modes
  - ARM processor power modes
  - Processor standby mode
  - Standby use cases and considerations
  - Using WFE to enter standby
  - Processor power down
  - Power down example
  - Barriers and power down modes
  - Entering power modes
  - Point of no return

- Multiprocessor and System Power Modes
  - Multiprocessor power modes
  - L2 cache power considerations
  - Power down mode examples
  - Preparing the L2 cache for power down
  - SoC and system power down

❖ Debug

- Debug Overview
  - Why debug?
  - Types of debug (invasive/non-invasive)
  - How close to reality?

- Invasive Debug
  - GDBServer vs Bare metal
  - Debug infrastructure
  - How do I access debug logic?
  - Debug registers
  - Debugger invasiveness
  - Debug events
  - Halt vs Monitor mode debugging

- Viewing memory
- Debugger impact on caches
- Vector catch
- Instruction breakpoint types
- Breakpoint comparison
- Embedded cross trigger- CTI
- Debugger semi-hosting support

- Non-Invasive Debug (PMU and Trace)
  - Performance monitoring hardware
  - PMU configuration for Cortex-A
  - PMU example code
  - Are my numbers meaningful?
  - A CoreSight trace system
  - Trace introduction
  - Other trace sources
  - Trace sinks (ETB, TPIU)

❖ **NEON Overview**

- NEON Introduction
  - What is NEON?
  - Example SIMD instruction
  - Why program for NEON?
  - Power considerations
  - NEON registers
  - NEON hardware details
  - Floating point
  - Enabling NEON in software
  - NEON status registers

- NEON Instruction Set Overview
  - Instruction syntax
  - Instruction modifiers
  - Instruction shapes
  - NEON data types
  - Specifying data types
  - NEON sample instructions

- NEON Software Support
  - How to use the NEON coprocessor
  - What is Project Ne10?
  - What is OpenMAX?
  - Automatic vectorizing
  - Tuning C/C++ code for vectorizing
  - NEON vectorizing example
  - Intrinsics

❖ **TrustZone**

- TrustZone Overview
  - What is TrustZone?
  - Why do we need TrustZone?
  - What kind of attacks are there?
  - What does it add?
  - TrustZone is not…
- Moving Between Worlds
  - Moving between Normal and Secure worlds
  - Vector tables
  - Asynchronous exceptions – no trapping
  - Asynchronous exceptions – trap all
  - Exception handling example
  - Interrupt latency
- Memory System
  - Memory management
  - Secure and non-secure memory
  - Caches and TLBs
  - Example memory system
  - Security access violations
- Debug
  - Debug configuration
- Software
  - Booting and the Chain of Trust
  - Trusted services
  - How do you make use of a service?