RA1 - Cortex-A8 implementation

This course covers the Cortex-A8 high-end ARM core

Objectives

- This course is split into 3 important parts:
  - Cortex-A8 architecture
  - Cortex-A8 software implementation and debug
  - Cortex-A8 hardware implementation.
- MMU operation under Linux is described.
- Interaction between level 1 caches, level 2 cache and main memory is studied through sequences.
- The exception mechanism is detailed, indicating how virtualization enables the support of several operating systems.
- The course also details the hardware implementation and provides some guidelines to design a SoC based on Cortex-A8.
- An overview of the Coresight specification is provided prior to describing the debug related units.

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

Prerequisites and related courses

- Knowledge of ARM7/9 or having attended our course ARM fundamentals.
- This course does not include chapters on low level programming.
  - ACSYS offers a large set of tutorials to become familiar with RVDS, assembly level programming, compiler hints and tips.
- More than 12 correct answers to our Cortex-A prerequisites questionnaire.
- Related courses:
  - Programming with RVDS IDE,reference RV0 - Programming with RVDS IDEcourse
  - VFP programming, reference RC0 - VFP programmingcourse
  - NEON programming, reference RC1 - NEON-v7 programmingcourse

    .

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.
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Target Audience

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

Course Outline

First day

ARM BASICS

- States and modes
- Exception mechanism
- Instruction sets
- Purpose of CP15

TRUSTZONE

- TrustZone conceptual view
- Secure to non secure permitted transitions
- L1 and L2 secure state indicators, memory partitioning
- Boot sequence

INTRODUCTION TO CORTEX-A8

- Block diagram
- Highlighting the instruction path and the data path
- Supported instruction sets
- Exceptions
- Configurable options

INSTRUCTION PIPELINE

- Superscalar pipeline operation
- Studying how instructions are processed step by step
- Branch prediction mechanism, BTB and GHB usage
- Return stack
- Instruction Memory Barrier

MEMORY MANAGEMENT UNIT

- Page sizes
- Address translation
- Page access permission
- Page attributes
- Software vs hardware tablewalk
- TLB lockdown
- Abort exception
- MMU maintenance operations

Second day

CORTEX-A8 LEVEL 1 AND LEVEL 2 CACHES

- Cache basics
- L1 cache organization
- Hardware support for virtual aliasing conditions
- Write buffer
- L1 caches software read for debug purposes

- CP15 related registers
- L2 Cache organization
- Physical indexing, physical tagging
- L2 cache transfer policy
- Write buffer
- L2 Preload Engine [PLE], programming the channels
- L2 cache software read for debug purposes
- PMU related events
- CP15 related registers

AXI PROTOCOL

- PL301 AXI interconnect
- Separate address/control and data phases
- Support for unaligned data transfers
- Transaction ordering
- Read and write burst timing diagrams
- Cortex-A8 external memory interface, ID encoding

HARDWARE IMPLEMENTATION

- Clock domainsk
- Reset domains
- Power control, dynamic power management
- Wait For Interrupt architecture
- AXI master interface attributes
- Internal exclusive monitor, clarifying ldrex / strex instructions

Third day

PERFORMANCE MONITOR

- Event counting
- Selecting the event to be counted for the 4 counters
- Debugging a multi-core system with the assistance of the PMU

VECTORED INTERRUPT CONTROLLER

- Cortex-A8 exception management
- The 3 vector table base registers
- Interrupt virtualization
- Connection of an external interrupt controller
- Enabling interrupt nesting
- ARM PL192 VIC
- Sequence required to clear the interrupt source
- Cascading two PL192s

LOW POWER MODES

- Voltage domains
- Run mode, standby mode, dormant mode
- Studying the sequence required to enter and exit dormant mode
- Communication to the power management controller

CORESIGHT DEBUG UNITS

- Invasive debug, non-invasive debug

- APBv3 debug interface
- Debug facilities offered by Cortex-A8
- Process related breakpoint and watchpoint
- Program counter sampling
- Event catching
- Debug Communication Channel
- ETM interface, connection to funnel
- Cross-Trigger Interface, debugging a multi-core SoC