FCC2 - e5500 implementation

This course covers the e5500 core present in 64-bit QorIQ SoCs

Objectives

- This course provides a detailed description of the e5500 internal architecture as well as the associated low level routines.
- Coherency mechanisms required in multiple e5500 platforms are explained through sequences.
- All mechanisms required in a multiple core system are described: atomic sequence through lwarx/stwxc. instruction pair, doorbell interrupts.
- The course focuses on the benefits of the hypervisor: running several operating systems, partitioning, load balancing and virtualization.
- The operation of the MMU is studied, particularly the TLB software reload routines.
- The course details the interrupt proxy unit and provides guidelines to implement nesting.
- Note that for on-site course, the contents can be tailored to specific customer needs.

- This course has been designed in collaboration with NXP

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

Prerequisites

- Experience of a 32-bit processor or DSP is mandatory.

Exercise: The environment used to build and debug software labs are based on the GNU compiler / linker and the debugger from Lauterbach.

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

e5500 CORE OVERVIEW

- Highlighting data path and instruction path
- Changes from e500mc to e5500

e5500 HYPERVISOR STATE

- Processor privilege levels state machine, user, guest OS, hypervisor
- Bare-metal operation
- Collaboration between guest OS and hypervisor to reload TLBs

- Directed interrupts
- Messaging within a coherency domain
- Filtering incoming messages

PIPELINE

- Instruction pipeline operation, dual issue, out of order execution
- Issue queue resource requirements
- Dispatch conditions, completion conditions
- Execution and context serializations, purpose of the isync instruction
- Branch management: dynamic prediction, BTB
- Link stack
- Segment target index cache (STIC) and segment target address cache (STAC)
- Guarded memory

DATA AND INSTRUCTION PATHS

- Implementation of a spin lock routine
- Decorated storage facility
- Memory barriers
- List insertion in a multicore system

COMPUTATION MODES

- Selecting 32-bit thread mode or 64-bit thread mode
- Computing effective addresses
- 64-bit arithmetic instructions

FLOATING POINT UNIT

- FPU operation: FPSCR register, IEEE vs non-IEEE mode
- Float load / store instructions
- Float arithmetic instructions
- Convert instructions
- Fully pipelined FPU

THE EXCEPTION MECHANISM

- Exception management: building the handler table through IVPR,IVOR registers
- Recoverable vs non recoverable exceptions
- Requirements to support exception nesting
- Exception priorities
- Interrupt proxy
- Multicore exceptions, doorbells and messages
- Integrated timers
- Reset sequence, initialization requirements

THE MEMORY MANAGEMENT UNIT

- MMU objectives definition
- Address translation, understanding the interim 48-bit virtual address
- Process protection through TID
- Two-level MMU architecture, level-1 TLBs and level-2 TLBs
- TLB organization, TLB software management, MAS registers
- Software TLB reload, clarifying the hardware assistance to select the victim in L2TLB0
- Managing a page descriptor table in a SMP system, tlbivax instruction
- Virtualization fault, managing the MMU at hypervisor level
- External PID load and store instructions

- TLB parity protection, multiple-hit detection

L1 AND L2 CACHES, SNOOPING

- e5500 L1 cache
- L2 cache organization
- Hit under miss and miss under miss
- Store miss merging
- Dynamic Harvard implementation
- Write shadow mode
- MESI snooping sequences involving two e5500 and a PCI Express master
- Data & instruction prefetch instructions
- Cache entry locking
- Stashing capability
- L1 and L2 error checking and correction, error injection

DEBUG

- Performance monitor
- Nexus debug unit
- Instruction and data breakpoints, programming address ranges
- Debug data acquisition message
- Debug Notify Halt instruction
- Nexus trace

POWER MANAGEMENT

- Connection to platform PM unit
- Power states
- Wake-up interrupt