PG2 - PowerPC System Design

This course has been designed for developers involved in a PowerPC development who want to understand generic mechanisms

Objectives

- This course explains the objectives of mechanisms used to boost the performance and the way they are implemented in various PowerPCs : cache / cache coherency, pipeline, MMU, exceptions.
- This gives to the attendees a wider overview of the state of the art in these domains.
- The course details the instructions required to write program in supervisor mode to adapt the behaviour of the core to specific needs.
- Task switch requirements are highlighted.
- Debug facilities implemented in PowerPCs (hardware breakpoints, real-time trace, watchpoints) are studied through the use of Lauterbach TRACE32 debugger.

A lot of programming examples have been developed by ACSYS to explain the PowerPC assembly language.
• They have been developed with GNU compiler and are executed under Lauterbach debugger.
A more detailed course description is available on request at training@ac6-training.com

Prerequisites

- Basic knowledge of processor or DSP.

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

PowerPC PROGRAMMING

- PowerPC programming environment : 32-bit PowerPC architecture, Book E, 64-bit architecture
- Register set, GPR vs SPR, HID registers
- Data type instantiation for PowerPC
- Pointers management (Addressing modes)
- User and supervisor functions call and return (EABI, C-to-assembly interface)
- Sections, benefits of small data sections
- Locating code and data in memory , linker command file
- Reset, what is to be done before calling the main() : Cstart program

PIPELINE

- Superscalar operation
- Mechanisms used to boost performance : branch prediction, branch target address cache, link stack
- Guidelines to optimize execution time
- Serializations, isync instruction, determining when this instruction is really required

DATA PATH AND DECOUPLING

- Highlighting the frequency domains present in PowerPC : core and bus interface
- Decoupling the core from cache and bus through load and store buffers
- Default ordering of load and store transactions
- Enforcing the ordering through eieio (called mbar in Book E) and sync (called msync in Book E) instructions
- Purpose of the Guarded attribute
- Consequence for high level development of IO drivers

MEMORY MANAGEMENT UNIT

- Requirements for kernels enabling dynamic memory mapping
- Single process multi-thread versus multiprocess multi-thread kernels
- Objectives of the MMU : page protection, definition of page attribute, address translation
- Segment and page translation
- Table search mechanisms : benefits of a software table search
- Operation of TLB caches
- TLB programming, static initialization

CACHE AND DATA COHERENCY

- Introduction to cache memory
- Cache organization
- Write policies
- Replacement algorithms
- Data flow between external main memory
- Distinguishing private memory that is accessed only by the core and shared memory that can be accessed by the core and other masters (DMA or CPU)
- Software enforced coherency
- Hardware enforced coherency

EXCEPTION MECHANISM

- Software exceptions vs interrupts
- Save / restore registers
- Organization of an exception handler : prolog, body and epilog
- How to find the cause of the exception, syndrome registers
- Design of a generic exception handler based on a vector table
- Interrupt management, addition of a critical interrupt in Book E
- Integrated interrupt controller
- Requirements for interrupt nesting

MULTITASK

- Management of boolean semaphores, lwarx / stwcx. instruction pair
- Stack switch, use of SPRG registers
- Definition of the set of registers that determine the stack state
- Management of task lists in single and multi processor systems

PowerPC DEBUG SOLUTIONS

- On-chip debug logic
- Restrictions of JTAG debug

- Hardware breakpoints
- Real-time trace
- Debugging software when caches are active
- The performance monitor