RM4 - Cortex-M7 implementation

This course covers the Cortex-M7 V7E-M compliant CPU

Objectives

- This course is split into 3 important parts:
  - Cortex-M7 architecture
  - Cortex-M7 software implementation and debug
  - Cortex-M7 hardware implementation.
- Through a tutorial, the Cortex-M7 low level programming is explained, particularly the ARM linker parameterizing and some tricky assembly instructions.
- Note that attendees can replay these labs after the training.
- The course also indicates how to use caches and TCMs which are new units with regard to Cortex-M4.
- The course also details the hardware implementation and provides some guidelines to design a SoC based on Cortex-M7, taking benefit of concurrent AXI/AHB transactions.
- 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 training@ac6-training.com

Prerequisites and related courses

- Our course reference RI0 - AXI3 / AXI4 INTERCONNECTcourse details the operation of AXI4 bus.
- Contents can adapted when attendees have already the knowledge of Cortex-M4.

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.

Evaluation modalities

- The prerequisites indicated above are assessed before the training by the technical supervision of the traineein his company, or by the trainee himself in the exceptional case of an individual trainee.
- Trainee progress is assessed by quizzes offered at the end of various sections to verify that the trainees have assimilated the points presented
- At the end of the training, each trainee receives a certificate attesting that they have successfully completed the course.
  - In the event of a problem, discovered during the course, due to a lack of prerequisites by the trainee a different or additional training is offered to them, generally to reinforce their prerequisites,in agreement with their company manager if applicable.

Plan

FIRST DAY - ARCHITECTURE

INTRODUCTION TO ARM CORTEX-M7

- Memory interfaces: TCMs, caches and AHB peripheral port
- Fixed memory map
- Configuration options
- Lock-step implementation

ARM CORTEX-M7 CORE

- Highlighting the new features of the V7E-M architecture
- In-order superscalar pipeline
- Dynamic branch prediction
- Bit-banding
- System timer
- System control block
- Detail of Data Processing Unit
- Load Store Unit, store buffering

CACHES AND TCMS

- Caches
  - Cortex-M7 cache implementation
  - Cache maintenance operations
  - Dynamic read allocate mode, recovering from errors
  - Store buffer merging
  - L1 Caches error correcting code
- TCMs
  - Benefit of implementing two separate D-TCM ports
  - Implementing external ECC for TCMs

SECOND DAY - ARCHITECTURE

ARCHITECTURE OF A SOC BASED ON CORTEX-M7

- Internal bus matrix
- Accessing ROM
- Accessing SRAM
- Connecting peripherals
- Sharing resources between Cortex-M7 and other CPUs
- STM32F7 architecture

EXCLUSIVE RESOURCE MANAGEMENT AND LOW POWER MODES

- Atomicity in single processor multiple thread systems
- Operation of the Local monitor
- Wait For Interrupt
- Events

EXCEPTIONS

- Exception behavior

- Exception-continuable instructions
- Non-maskable exceptions
- Fault handling MPU faults, external faults
- Priority boosting
- Reset sequence, initialization requirements

INTERRUPTS

- Interrupt entry / exit, timing diagrams
- Tail chaining
- NVIC registers
- Interrupt prioritization
- Interrupt handlers
- Wake-up Interrupt Controller

THIRD DAY – OPTIONAL UNITS, HARDWARE IMPLEMENTATION

MEMORY PROTECTION UNIT

- Device and normal memory ordering
- Memory type access restrictions
- Memory ordering restrictions
- Memory protection overview, ARM v7 PMSA
- Fault status and address registers
- Region overview, memory type and access control, sub-regions
- Region overlapping
- Setting up the MPU

FLOATING POINT UNIT

- Introduction to IEEE754,
- Floating point arithmetic
- Cortex-M7 single and double precision FPU
- Hardware support for denormals and all IEEE rounding modes
- Improving the performance by selection flush-to-zero mode and default NaN mode
- Extension of AAPCS to include FP registers
- Lazy floating-point context save
- Highlighting the new features of FPv5

AXI IMPLEMENTATION

- Overview of AXI bus specification, explaining ordering rules
- AXI attributes and transactions
- Restrictions on AXI transfers

AHB IMPLEMENTATION

- Overview of AHB-Lite protocol
- AHBP
- AHBS

AHB CORTEX-M7 HARDWARE IMPLEMENTATION

- Pinout
- Clocking and reset, power management
- Using an external Wake-up Interrupt Controller (WIC)

L2C-310 LEVEL 2 CACHE

- Cache configurability
- AXI interface characteristics
- Understanding through sequences how cacheable information is copied from memory to level 1 and level 2 caches
- Transient operations, utilization of line buffers LFBs, LRBs, EBs and STBs
- Power management
- Cache event monitoring
- Cache lockdown
- Interrupt management

FOURTH DAY – DEBUG, SOFTWARE DESIGN

INVASIVE DEBUG

- Coresight debug infrastructure, DAP
- Cortex-M7 debug features
- Halt mode
- Vector catching
- Monitor mode
- Debug event sources
- Flash patch and breakpoint features
- Data watchpoint and trace
- AHB-lite Debug interface
- ROM table

NON-INVASIVE DEBUG

- Basic ETM operation
- Instruction trace principles
- Instrumentation Trace Macrocell
- DWT trace packets
- Time-stamping packets
- Instruction tracing, branch packets, exception tracing packets
- TPIU components
- Embedded Trace Buffer

CROSS-TRIGGER INTERFACE

- Purpose of this debug unit
- Trigger signals to CTI and Trigger signals from CTI

EMBEDDED SOFTWARE DEVELOPMENT WITH CORTEX-M7

- Embedded development process
- Application startup
- Placing code, data, stack and heap in the memory map, scatterloading
- Reset and initialisation
- Placing a minimal vector table
- Further memory map considerations, 8-byte stack alignment in handlers
- Building and debugging your image
- Long branch veneers
- Coding guidelines when a cache is used

C/C++ COMPILER HINTS AND TIPS FOR Cortex-M7

- ARM compiler optimisations, tail-call optimization, inlining of functions
- Mixing C/C++ and assembly
- Coding with ARM compiler
- Measuring stack usage
- Unaligned accesses

- Local and global data issues, alignment of structures
- Further optimisations, linker feedback

THUMB-2 INSTRUCTION SET

- Data processing instructions
- Branch and control flow instructions
- Exception generating instructions
- If…then conditional blocks
- Stack in operation
- Memory barriers and synchronization

CORTEX-M7 DSP INSTRUCTION SET

- Multiply instructions
- Packing / unpacking instructions
- V6 ARM SIMD packed add / sub instructions
- SIMD combined add/sub instructions
- Multiply and multiply accumulate instructions
- SIMD sum absolute difference instructions
- SIMD select instruction
- Saturation instructions

Renseignements pratiques

Duration : 4 days
Cost : 2740 € HT